LOS 20 MEJORES PROGRAMAS PARA EL ZX SPECTRUM

indescomp

Los 20 mejores Programas para el ZX Spectrum

por ANDREW HEWSON
traducido por FEDERICO SANCHEZ-VALLEJO

Segunda Edición revisada Junio 1984
Copyright® (DERECHOS CEDIDOS A INDESCOMP)

Reservados todos los derechos. Ninguna parte de esta publicación puede ser reproducida o transmitida por cualquier forma o manera, tanto electrónica como mecánica, incluyendo fotocopiado, grabación o sistema de almacenamiento y recuperación, sin permiso por escrito de la Editorial.

Los programas de este volumen son propiedad de la Editorial y se destinan al uso personal y privado de los compradores.

I.S.B.N.: 84-86176-06-9

Composición de texto y programas e Impresión: CONORG, S.A.

Depósito Legal: M-22728-1984

AGRADECIMIENTO

Ningún libro de esta naturaleza es la obra de una sola persona. Agradezco a mi hermano GORDON por su lealtad y apoyo, y a LOUISE ESPLIN por descifrar y mecanografiar mi manuscrito y, principalmente, agradezco a JANET el haber asumido la mayor parte del trabajo, debido a lo cual pude disponer de tiempo para completar mi tarea.

CONTENIDO Página

0 Prólogo 7

I Introducción 9

II Programas de uso corriente

1. FICHERO 11
2. DIARIO 20
3. CALENDARIO 27

III Programas divertidos

4. AHORCADO 31
5. FUTBOL 35
6. MUSICA 38

IV Programas estadísticos y matemáticos

7. REPRESENTACIÒN DE DATOS 44
8. REGRESION 51
9. HISTOGRAMA 57
10. DIBUJO DE UNA FUNCION 62

V Programas útiles

11. RENUMERADOR DE LINEAS 66
12. CARGA DE RUTINAS EN CODIGO MAQUINA (HEX LOADER) 69
13. DESENSAMBLADOR 76
14. EDITOR DE CODIGO MAQUINA 85

VI Programas para el manejo de pantalla

15. GRAFICOS 95
16. BOCETOS 103
17. ESPIRALES 109
18. LA CAZA DEL PATO 116
19. DIBUJO (DIGITISER) 122
20. CALEIDOSCOPIO 128

PROLOGO

El ZX SPECTRUM ha sido la gran novedad del año 82 en el Reino Unido dentro de los ordenadores personales. Y lo va a ser en España a partir de la primavera de 1983. Este pronóstico puede parecer algo presuntuoso o mágico, entrando en el terreno de lo adivinatorio, pero no lo es.

Las ventajas del ZX SPECTRUM sobre el ZX 81, verdadero impacto en su tiempo, le hacen igualmente deseable que aquél pero dentro de una perspectiva mucho más avanzada.

Así, el amante de las matemáticas no necesitará ya crear subrutinas para cada una de las funciones que quiera definir, y podrá representarlas mediante gráficos de alta resolución con la precisión que desee.

El economista podrá hacer uso de programas de datos mediante la opción MERGE, evitando la incorporación de datos dentro del propio programa, siempre tan farragosa.

El aficionado a los juegos puede ya estar satisfecho: aquí se aúnan todas las nuevas características del SPECTRUM: color, alta resolución, dibujo sobre dibujo, etc. para llegar a programas suficientemente completos e interesantes para todas las edades.

Las características de capacidad de memoria en 16 y 48K y la posibilidad de mezcla de dibujo y texto en gráficos de alta resolución hacen del SPECTRUM una herramienta ideal para la enseñanza individualizada y en la propia casa del alumno, especialmente indicada en relación con las representaciones gráficas.

Como no todo puede ser perfecto también existen lagunas en el ZX SPECTRUM, sobre todo en el área de sonidos (poco desarrollada).

ANDREW HEWSON ha desarrollado el presente libro mediante una complejidad paulatina a través de 20 programas, todos ellos interesantes tanto en razón de la utilidad futura como del aprendizaje práctico de las características del ZX SPECTRUM.

El autor comienza analizando la creación de ARCHIVOS (3 primeros programas), continúa con DIBUJO y MUSICA, pasa por aplicaciones estadísticas y CODIGO MAQUINA, y termina con dibujos complejos, manteniendo siempre un equilibrio envidiable entre programas en BASIC y subrutinas en código máquina.

En definitiva, consideramos que es un BUEN LIBRO, que debe ser BASICO en la biblioteca de todo usuario del ZX SPECTRUM.

FEDERICO SANCHEZ-VALLEJO

INTRODUCCION

En 1980 la SINCLAIR RESEARCH LTD. lanzó al mercado el primer computador por debajo de la barrera de las 100Ł, el ZX80. Con sólo 1K de RAM incorporado, variables enteras y unos resultados en pantalla que desaparecían durante el cálculo, el ZX80 era un pariente pobre de los otros microcomputadores disponibles por aquel entonces. Sin embargo, desplazó del mercado a la cuarta parte de sus competidores y se difundió tan rápidamente que los fabricantes pronto le proclamaron el mejor computador británico.

En la primavera de 1981 SINCLAIR lanzó un sustituto, el ZX81. Este computador tenía todavía 1K de RAM incorporada, pero utilizaba variables en coma flotante y podría mantener los resultados en la pantalla durante el cálculo a voluntad del usuario (aunque a riesgo de conseguir una menor velocidad en el cálculo). El precio del ZX81 era un poco inferior a 70Ł.

Ahora SINCLAIR lanza su computador de 1982, el ZX SPECTRUM. Tiene incorporadas 16K de RAM en su versión standard, incluye gráficos en color y alta resolución (256 x 176 puntos) y dispone de un software más perfeccionado que el ZX81, incluyendo la posibilidad de conexión de un disco muy manejable. Cuesta 125Ł (versión 16K).

Las mejoras del ZX80 y, aún más, del ZX81, no se echan en saco roto y siguen desempeñando el papel de una primera etapa para los usuarios, los cuales desean explorar el mundo de los computadores de la manera más barata posible. Se necesita algo de ingenio para obtener buenos resultados de estas máquinas -lo que no es mala cosa al no tener una compensación de tipo económico por sus esfuerzos.

Por esta razón yo escribí 2 libros: "HINTS & TIPS FOR THE ZX80" y más tarde "HINTS & TIPS FOR THE ZX81" (Indicaciones y consejos para el ZX80 y el ZX81), los cuales intentaban proporcionar al lector algunos conocimientos sobre el MODUS OPERANDI de estos computadores.

Para escribir este libro sobre el ZX SPECTRUM he adoptado una actitud algo diferente. Con 16K de RAM en su versión standard la presión sobre el usuario para escribir programas con economía de espacio (de memoria) es mucho menor. Recíprocamente, al disponer de una amplia gama de comandos y funciones, existe una mayor motivación para que el usuario escriba largos y más sofisticados programas.

El escribir buenos programas es una habilidad que puede ser aprendida por el usuario, y mi propósito en este libro es enseñarla mediante ejemplos. Mis programas ilustran un número de técnicas que son bien conocidas en el mundo de los computadores- el uso de registros de longitud fija y variable; la clasificación por burbuja; la búsqueda doble (DIARIO y FICHERO).

Otros programas ilustran técnicas peculiares del SPECTRUM poniendo énfasis en la utilización más eficiente de los resultados. La mayor parte de los programas están divididos en varias secciones, comenzando cada una de ellas con una línea múltiplo de 500, pudiendo así el usuario progresar adecuadamente y adaptar los programas a sus propias necesidades.

Espero que el lector no intente digerirlo de principio a final y de una sola vez. Más bien debe elegir un apartado, leerlo, dejar volar su fantasía y trabajar con el programa que aquí listamos. Yo he descrito los aspectos del programa que a mí me parecen más importantes; en algunos casos con gran detalle y en otros, tratados con anterioridad en este libro, de forma más bien superficial.

El escribir un libro como éste es una tarea que lleva mucho tiempo. Así que espero que el lector disfrute utilizando los programas como yo he disfrutado escribiéndolos.

ANDREW HEWSON
BLEWBURY 1982

1. FICHERO

Este es un programa de tipo general para almacenamiento de datos.

Mediante el MENU de opciones se permite al usuario establecer la naturaleza de los datos que van a ser almacenados, añadir y suprimir datos, visualizar los datos en la pantalla o en la impresora, buscar los datos deseados, ordenarlos tanto de forma numérica como alfabética y grabarlos (SAVE) y cargarlos (LOAD) mediante un cassette. El nombre de **FICHERO** es debido a que está diseñado para que funcione como un sistema de fichas, pero con una gran velocidad, flexibilidad y comodidad de utilización.

El usuario controla el programa introduciendo un solo comando de su elección (letra) conforme a lo indicado en el MENU (línea 200); ilustrado en la figura FN1. El código del carácter elegido es almacenado en la cadena Z$ y se compara con cada uno de los códigos posibles en la línea 230.

Si son iguales el valor del paréntesis es 1. Un símbolo de multiplicación previo a cada paréntesis dirige el programa a la rutina elegida. Así si el usuario pulsa "H" para PARAR, el paréntesis final de la línea 230 formará el valor 1 puesto que Z$ = "H" es verdad. Como todos los otros paréntesis son CERO la línea 230 se leerá como GOTO 300 + 4200. El programa salta a la línea 4500, escribe (PRINT) un mensaje y se para (STOP).

```
                    FICHERO

              Menu de Opciones

Definicion estructura ficha   D
Anadir fichas                 A
Borrar fichas                 O
Revisar fichas                R
Imprimir fichas               P
Buscar fichas                 T
Clasificar fichas             B
Grabar fichero en cassette    S
Cargar el            fichero  L
Verificar el fichero          V
Parar el programa             H
```

FIGURA FN1. **El MENU de opciones del programa FICHERO**

Si el usuario introduce un código no existente en el MENU, el programa salta a la línea 300, realiza una PAUSA y retorna al MENU.

Cada una de las opciones del MENU se ejecuta de forma más o menos independiente de las otras, para lo cual el programa se divide en módulos, uno para cada opción. Por convenio, el mismo de la primera línea de cada módulo es un múltiplo de 500.

CREACION DE LA ESTRUCTURA DE UNA FICHA — INTRODUCIR D

Vamos a suponer que el programa se utiliza para mantener los datos de un CLUB de MICROCOMPUTADORES. Se introducen como datos **a)** el nombre, **b)** la inicial de cada miembro, **c)** el ordenador que posee, **d)** la cuota mensual que paga y **e)** si está al corriente de los pagos o no.

¿Cómo podría almacenar esta información el FICHERO?

Lo primero que debemos hacer es establecer una estructura variable para almacenar los datos conforme a la opción "DEFINICION DE LA ESTRUCTURA DE UNA FICHA". El programa salta a la línea 1000 y pregunta al usuario cuál es el número total de FICHAS a introducir. Se requerirá una ficha para cada uno de los miembros del club y así podríamos necesitar, por ejemplo, 50 fichas para los 50 miembros. El programa pregunta a continuación por el número de campos literales (*) y el número de campos numéricos. Sería muy conveniente que organizáramos los 5 conceptos a almacenar (nombre, inicial y ordenador que posee el miembro, cuotas pagadas o no) mediante 4 campos literales. (La cuota mensual es el único campo numérico).

El programa pregunta a continuación el nombre de cada uno de los campos de forma ordenada y, en el caso de campos literales, por el máximo número de caracteres que pueden ser almacenados en cada uno de ellos. La longitud del "NOMBRE del campo" puede ser de 10 caracteres o menos, pero no hay límite para la longitud del campo.

Una estructura apropiada de FICHA se incluye en la Figura FN2.

NUMERO MAXIMO DE FICHAS	=	50
NUMERO DE CAMPOS LITERALES	=	4
NUMERO DE CAMPOS NUMERICOS	=	1

NUMERO DEL CAMPO LITERAL	**NOMBRE**	**LONGITUD**
1	MIEMBRO	20
2	INICIAL	1
3	COMPUTADOR	20
4	PAGOS ? (S/N)	1

NOMBRE DEL CAMPO NUMERICO	**NOMBRE**
1	CUOTA

FIGUNA FN2. Un ejemplo de la estructura de un FICHERO. Esta estructura está dimensionada para almacenar el nombre, la inicial, el computador y detalles acerca de las cuotas para los 50 miembros de un Club de Computadores.

(*) NOTA DEL TRADUCTOR: Compuestos por cadenas, tanto alfabéticas como numéricas.

AÑADIR FICHAS — INTRODUCIR A

Esta opción se utiliza para introducir datos uno detrás de otro inmediatamente después de creada la estructura de las FICHAS o en una fecha posterior. El programa escribe el número de la primera ficha vacía e indica el dato a introducir en ese momento. Si se comete un error debemos pulsar "B" y retrocedemos al campo anterior. En el caso de pulsar "F" (la entrada de datos se ha terminado) se vuelve AL MENU de opciones.

BORRAR FICHAS — INTRODUCIR O

Esta opción se utiliza para borrar fichas. El programa pregunta por los números de las fichas que son comienzo y final del borrado. Para borrar, por ejemplo, las 3 primeras fichas se han de introducir, primero un 1 y luego un 3. Se ha de tener cuidado al borrar más de una ficha de una sola vez, ya que el programa produce fallos debido a la RENUMERACION de las fichas. La manera más fácil de evitar problemas es borrar primero el grupo cuyo número de orden es el mayor. Por ejemplo: Si se han de borrar las fichas 3 a 7 y 14 a 16 borraremos primero el segundo grupo (14 a 16) y luego el primero (3 a 7).

REVISAR O IMPRIMIR ALGUNA FICHA — INTRODUCIR R o P

El programa, pregunta por el número de las fichas en las cuales se desea comenzar y terminar la operación y dichas fichas aparecen en pantalla o en la impresora.

BUSCAR FICHAS — INTRODUCIR T

El programa pregunta por los números de las FICHAS en que se desea comenzar y terminar la búsqueda y el nombre del campo que se busca. El programa entonces compara el nombre introducido por el usuario con el contenido del campo en cada una de las FICHAS de forma ordenada y, si se encuentra uno idéntico, aparece en pantalla el número de dicha FICHA.

CLASIFICACION DE FICHAS POR BURBUJA — INTRODUCIR B

El programa pregunta por el número del campo que señala el orden en el cual se han de clasificar las fichas. Si el campo es literal la clasificación se realiza por el CODIGO de los caracteres de cada concepto dentro de cada campo. Si el campo es numérico las fichas se ordenan por orden numérico.

GRABACION (SAVE), CARGA (LOAD) y VERIFICACION (VERIFY) — INTRODUCIR S, L o V

Estas opciones permiten al usuario grabar los datos en cassette, cargar los datos grabados en el cassette o verificar si la copia contenida en un cassette es idéntica al contenido de la memoria.

MAS ACERCA DEL PROGRAMA

Este FICHERO proporciona la mayor parte de las características requeridas de la Base de Datos de un computador, pero hay muchas más cosas con las que se le podría complementar. Por ejemplo, podría ser posible:

1. Editar los campos existentes en una ficha
2. Añadir nuevos campos a los conceptos ya existentes.
3. Buscar varios campos de forma simultánea.
4. Definir formatos especiales de salida de datos
5. Aumentar el número de fichas sobre el límite inicial definido
6. Crear nuevos conceptos mediante la clasificación secundaria de algunos casos.

La primera consideración sobre las características que podemos estudiar es la cantidad de memoria RAM disponible. El programa base, tal como está escrito, ocupa unas 6,5K de las 9K disponibles por el usuario en una máquina de 16K.

1. FICHERO

Teniendo en cuenta que los propios datos requieren una considerable cantidad de espacio (50 FICHAS de 20 caracteres cada uno, ocupan 1K, por ejemplo) es evidente que se requiere una máquina de 48K para poder añadir un software más completo al programa principal. La alternativa es dividir el programa en partes coherentes y añadir nuevos módulos a cada una de las partes. Cada nuevo programa debe poder grabar, cargar y verificar los datos para que cada uno de los programas pueda comunicarse con los demás.

Las variables más importantes utilizadas por el programa son:

nn	- Número de campos numéricos
ns	- Número de campos literales
cr	- Número de fichas utilizado
nr	- Número de fichas máximo
f$ (ns + nn, 10)	- Cadena literal (conteniendo los nombres de los campos)
lr	- Longitud total de todos los campos literales
b$ (ns, lr)	- Cadena literal conteniendo toda la información de las cadenas literales
b (ns, 2)	- Cadena numérica conteniendo las direcciones en b$ de comienzo y terminación de cada campo literal
a (nr, nn)	- Cadena numérica conteniendo toda la información numérica
c (4)	- Cadena numérica en la que se almacenan **nn, ns, cr** y **nr** cuando se GRABAN en cassette.

El último concepto de interés es la CLASIFICACION por BURBUJA. Este método de clasificación puede parecer ineficaz, pero de hecho es mucho más rápido que otras técnicas que aparentan ser más sofisticadas.

En una clasificación por burbuja el ordenador verifica los datos desde el principio hasta el final, comprobando si cada nuevo dato es mayor o igual que su predecesor. Sólo si es menor las posiciones entre los 2 últimos datos se intercambian. El computador repite el proceso hasta que realiza una pasada a través de todos los datos sin realizar ningún cambio, momento en el que la clasificación está completa. Durante la clasificación, los datos "menores" ascienden un escalón cada vez, como si fueran BURBUJAS que borbotearan, motivo por el que se da dicho nombre a esta técnica. El método es muy popular porque es fácil de programar, rápido de ejecutar y no requiere demasiado espacio, excepto la variable que se utiliza para indicar la duración de cada paso cuando se ha producido un intercambio entre las posiciones de 2 datos.

CODIGOS DE ERROR	SIGNIFICADO
1	La letra cuyo código se ha introducido no figura en el MENU.
2	Intento de establecer una estructura de FICHA absurda, p.e.: Sin FICHAS o campos de números negativos.
3	Nombre de campo mayor de 10 caracteres.
4	Cadena con longitud menor que 1.
5	Nombre de FICHERO mayor de 10 caracteres.
6	Intento de buscar un campo inexistente.

PROGRAMA PN1. FICHERO

```
  10 LET nn=0
  20 LET ns=0
  30 LET t=28
  40 LET cr=0
 200 CLS : PRINT TAB 9;"FICHERO",,,TAB 5;"Menu de Opciones",,,
     "Definicion estructura ficha";TAB t;" D","Anadir fichas":
     TAB t;" A","Borrar fichas";TAB t;" O",
 202 PRINT "Revisar fichas";TAB t;" R","Imprimir fichas";TAB t;
     " P","Buscar fichas";TAB t;" T","Clasificar fichas";TAB t;
     " B","Grabar fichero en cassette";TAB t;" S","Cargar el
     fichero";TAB t;" L","Verificar el fichero";TAB t;" V",
     "Parar el programa";TAB t;" H"
 210 INPUT z$
 220 GO SUB 6000
 230 GO TO 300+5200*(z$="T")+4700*(z$="O")+200*(z$="A")+700*
     (z$="D")+1200*(z$="R")+1700*(z$="P")+2200*(z$="B")+2700*
     (z$="S")+3200*(z$="L")+3700*(z$="V")+4200*(z$="H")
 300 GO SUB 6050: PRINT "1"
 310 PRINT ,,"Pulsa una tecla para continuar"
 320 PAUSE 0
 330 GO TO 200
 510 PRINT ,,"Da valor para cada uno de los campos de cada ficha"
     ,,;TAB 0;"Introduce:",,;TAB 0;" F para terminar";TAB 0;
     " B para volver un espacio atras"
 520 FOR i=cr+1 TO nr
 530 PRINT "Ficha no";i
 540 FOR J=1 TO ns+nn
 550 PRINT " Campo ";f$(j);
 560 INPUT z$
 570 IF z$="F" OR z$="f" THEN  PRINT ,,,"Fin de entrada de datos":
     GO TO 200
 580 IF J<>1 AND (z$="B" OR z$="b") THEN  LET j=j-1: GO TO 550
 590 IF J=1 AND z$="B" THEN  LET i=i-1: LET cr=cr-1: LET J=ns+nn:
     PRINT ,,,"Ficha no";i: GO TO 550
 600 IF J<=ns THEN  LET b$(i,b(J,1) TO b(J,2))=z$: GO TO 670
 610 LET a(i,J-ns)=VAL z$
 670 PRINT z$
 680 NEXT J
 690 LET cr=cr+1
 700 NEXT i
 710 PRINT "Fichero completo"
 720 GO TO 310
1000 PRINT ,,,,;"Creacion de estructura de ficha",,;TAB 0;
     "Introduce :";TAB 0;"No. max de fichas";TAB 0;"No. de campos
     literales";TAB 0;"No. de campos numericos",,,
1010 INPUT nr: PRINT AT 18,25;nr;
1012 INPUT ns: PRINT AT 19,25;ns;
1014 INPUT nn: PRINT AT 20,25;nn
1015 PRINT
1020 LET nr=INT nr: LET ns=INT ns: LET nn=INT nn
1030 IF nr<=0 OR ns<0 OR nn<0 OR ns+nn=0 THEN  GO SUB 6050:
     PRINT "2": GO TO 310
1040 DIM f$(ns+nn,10)
1050 LET lr=0
1060 IF ns=0 THEN  GO TO 1250
1070 DIM b(ns,2)
1110 FOR i=1 TO ns
1120 PRINT "Campo literal  ";i
```

```
1130 PRINT ,,"Nombre del campo?"
1140 INPUT z$: IF LEN z$>10 THEN  GO SUB 6050: PRINT TAB 25;"3":
     GO TO 1130
1150 LET f$(i)=z$
1160 PRINT AT 19,25;z$
1170 PRINT AT 21,0;"Longitud?";TAB 20
1180 INPUT l: IF l<1 THEN  GO SUB 6050: PRINT TAB 25;"4":
     GO TO 1170
1190 PRINT AT 21,25;l: LET b(i,1)=lr+1: LET lr=lr+l: LET b(i,2)=lr
1195 PRINT ,,
1200 NEXT i
1210 DIM b$(nr,lr)
1220 IF nn=0 THEN  GO TO 1340
1250 FOR i=ns+1 TO ns+nn
1260 PRINT "Campo numerico ";i
1265 PRINT
1270 PRINT "Nombre del campo? "
1280 INPUT z$: IF LEN z$>10 THEN  GO SUB 6050: PRINT TAB 25;"3":
     GO TO 1130
1290 LET f$(i)=z$
1300 PRINT AT 21,25;z$
1305 PRINT
1310 NEXT i
1320 DIM a(nr,nn)
1330 DIM c(nn)
1340 IF nn<4 THEN  DIM c(4)
1350 PRINT ,,"Estructura completa"
1360 GO TO 310
1500 PRINT ,,,TAB 9;"Revisar fichas"
1510 GO SUB 8500
1580 FOR i=b TO e
1590 PRINT "Ficha no. ";i
1600 FOR J=1 TO ns
1610 PRINT "  ";f$(J);" : ";
1620 PRINT b$(i,b(J,1) TO b(J,2))
1630 NEXT J
1640 FOR J=ns+1 TO ns+nn
1650 PRINT "  ";f$(J);" : ";
1660 PRINT a(i,J-ns)
1670 NEXT J
1680 NEXT i
1710 PRINT "Fin de la revision": GO TO 310
2000 PRINT ,,TAB 9;"Imprimir fichas"
2010 GO SUB 8500
2080 FOR i=b TO e
2090 LPRINT "Ficha no. ";i
2100 FOR J=1 TO ns
2110 LPRINT TAB 0;"  ";f$(J);" : ";
2120 LPRINT TAB 16;b$(i,b(J,1) TO b(J,2))
2130 NEXT J
2140 FOR J=ns+1 TO ns+nn
2150 LPRINT TAB 0;"  ";f$(J);" : ";
2160 LPRINT TAB 16;a(i,J-ns)
2170 NEXT J
2180 NEXT i
2200 PRINT ,,"Impresion terminada": GO TO 310
2500 PRINT ,,TAB 0;"Clasificacion de fichas"
2505 IF cr=0 THEN  PRINT "Archivo vacio": GO TO 310
2510 PRINT ,,;TAB 0;"Introduce No. de campo a clasi- ficar":
     INPUT fn
```

```
2520 LET fn=INT fn: IF fn<1 OR fn>ns+nn THEN  GO SUB 6050:
     PRINT "6": GO TO 2510
2530 IF fn>ns THEN  GO TO 2700
2540 LET in=0
2560 FOR i=1 TO cr-1
2570 FOR j=b(fn,1) TO b(fn,2)
2580 IF b$(i,j)<b$(i+1,j) THEN  GO TO 2660
2590 IF b$(i,j)>b$(i+1,j) THEN  GO TO 2620
2600 NEXT j
2610 GO TO 2660
2620 LET z$=b$(i)
2630 LET b$(i)=b$(i+1)
2640 LET b$(i+1)=z$
2645 IF nn>0 THEN  FOR j=1 TO nn: LET c(j)=a(i,j):
     LET a(i,j)=a(i+1,j): LET a(i+1,j)=c(j): NEXT j
2650 LET in=in+1
2660 NEXT i
2670 IF in>0 THEN  GO TO 2540
2680 PRINT ,,"Clasificacion terminada"
2690 GO TO 310
2700 LET in=0
2710 FOR i=1 TO cr-1
2720 IF a(i,nn)<=a(i+1,nn) THEN  GO TO 2800
2730 FOR j=1 TO nn
2740 LET c(j)=a(i,j)
2750 LET a(i,j)=a(i+1,j)
2760 LET a(i+1,j)=c(j)
2770 NEXT j
2780 IF ns>0 THEN  LET z$=b$(i): LET b$(i)=b$(i+1): LET b$(i+1)=z$
2790 LET in=in+1
2800 NEXT i
2810 IF in>0 THEN  GO TO 2700
2820 PRINT ,,"Clasificacion terminada"
2830 GO TO 310
3000 PRINT ,,TAB 0;"Grabacion del fichero",,"Introduce su nombre "
3010 INPUT z$: IF LEN z$>10 THEN  GO SUB 6050: PRINT "5":
     GO TO 3010
3020 LET c(1)=ns
3030 LET c(2)=nn
3040 LET c(3)=nr
3050 LET c(4)=cr
3060 PRINT AT 21,0; FLASH 1;"Grabando fichero"
3070 SAVE z$ DATA a()
3080 SAVE z$ DATA b()
3090 SAVE z$ DATA b$()
3100 SAVE z$ DATA c()
3110 SAVE z$ DATA f$()
3120 PRINT AT 21,0;z$;"Grabacion terminada"
3130 GO TO 310
3500 PRINT ,,TAB 0;"Cargar el fichero",,,"Introduce su nombre"
3510 INPUT z$: IF LEN z$>10 THEN  GO SUB 6050: PRINT "5":
     GO TO 3510
3520 PRINT AT 21,0; FLASH 1;"Cargando el fichero"
3530 LOAD z$ DATA a()
3540 LOAD z$ DATA b()
3550 LOAD z$ DATA b$()
3560 LOAD z$ DATA c()
3570 LOAD z$ DATA f$()
3575 CLS
3580 PRINT AT 11,0;z$;"Carga terminada"
3590 LET ns=c(1)
3600 LET nn=c(2)
3610 LET nr=c(3)
3620 LET cr=c(4)
3630 GO TO 310
```

```
4000 PRINT ,,TAB 0;"Verificar el fichero",,,"Introduce su nombre"
4010 INPUT z$: IF LEN z$>10 THEN  GO SUB 6050: PRINT "5":
     GO TO 3010
4020 PRINT AT 21,0; FLASH 1;"Verificando el fichero "
4030 VERIFY z$ DATA a()
4040 VERIFY z$ DATA b()
4050 VERIFY z$ DATA b$()
4060 VERIFY z$ DATA c()
4070 VERIFY z$ DATA f$()
4075 CLS
4080 PRINT AT 11,0;z$;" verificado el fichero"
4090 GO TO 310
4500 PRINT ,,"Se detiene el programa"
4510 STOP
5000 PRINT ,,TAB 10;"Borrar fichas"
5010 GO SUB 8500
5070 FOR i=1 TO cr-e
5080 IF ns<>0 THEN  LET b$(b+1-1)=b$(e+i)
5090 IF nn<>0 THEN  FOR j=1 TO nn: LET a(b+i-1,j)=a(e+i,j): NEXT j
5100 NEXT i
5110 LET cr=cr-e+b-1
5120 PRINT ,,"Fin del borrado"
5130 GO TO 310
5500 PRINT ,,TAB 9;"Buscar fichas"
5510 GO SUB 8500
5520 PRINT ,,"Nombre del campo a buscar"
5530 INPUT z$: IF LEN z$>10 THEN  LET z$=z$(1 TO 10)
5540 FOR i=1 TO ns+nn
5550 IF f$(i,1 TO LEN z$)=z$ THEN  GO TO 5600
5560 NEXT i
5570 GO SUB 6050: PRINT ;"6": GO TO 310
5600 PRINT ,,"Introduce la clave de busqueda"
5610 INPUT z$
5620 PRINT ,,z$;" aparece en las fichas :"
5630 IF i>ns THEN  GO TO 5700
5640 FOR J=b TO e
5650 FOR k=b(i,1) TO (b(i,2)-(b(i,2)-b(i,1)-LEN z$))*
     NOT (b(i,1)+LEN z$>b(i,2))
5660 IF z$=b$(J,k TO k+LEN z$-1) THEN  PRINT TAB 6;j: GO TO 5680
5670 NEXT k
5680 NEXT j
5690 GO TO 310
5700 FOR i=1 TO nn
5705 FOR j=b TO e
5710 IF a(j,i)=VAL z$ THEN  PRINT TAB 6;j
5720 NEXT j
5725 NEXT i
5730 GO TO 310
6000 LET z$=CHR$ (CODE z$-32*(CODE z$>96))
6010 RETURN
6050 BEEP .2,24
6055 PRINT
6060 PRINT "Error ";
6070 RETURN
6100 LET w=PEEK 23627+256*PEEK 23628
6110 RETURN
8500 IF cr=0 THEN  PRINT ,,"Archivo vacio": GO TO 310
8510 PRINT ,,"Introducir los Nos. de la prime-ra y de la
     ultima ficha"
8520 INPUT b,e
8530 LET b=1+(b-1)*(b>0)+(cr-b)*(b=cr)
8540 LET e=b+1+(e-b-1)*(e>b)+(cr-e)*(e=cr)
8550 RETURN
```

2. DIARIO

El programa precedente (FICHERO) asigna una cantidad prefijada de espacio para cada una de las variables de un campo dentro de cada registro. El campo asignado a cada "MIEMBRO" del club de microcomputadores, por ejemplo, ocupa 20 caracteres. En la mayor parte de los casos alguno de los caracteres podrían no ser utilizados con eficacia al tener el nombre de un miembro una longitud menor de 20 caracteres. Está claro que la técnica de almacenamiento de información en registros de longitud fija desperdicia una gran cantidad de espacio.

Yo he escrito el programa DIARIO para ilustrar la alternativa más simple que permite variar la longitud de los campos de las FICHAS tomadas de una en una. Los datos correspondientes a un día cualquiera constituyen una FICHA. Las características más importantes del programa son las siguientes:

1. Los datos pueden ser de cualquier longitud (y los llamaremos ANOTACIONES).
2. Sólo se asigna espacio para aquellos días en que se introduce algún dato.
3. Los datos correspondientes a un mismo día se almacenan automáticamente en la misma FICHA.
4. Los datos introducidos fuera de fecha se descartan automáticamente.
5. Se utiliza una doble técnica de búsqueda para localizar un dato dado.

ESTRUCTURA DEL PROGRAMA

La función de las principales variables del programa se muestra en la tabla TY1.

VARIABLES	CONTENIDO
tr	Número total de datos permitidos.
z$	Cadena literal utilizada para almacenar los datos.
tz	Longitud máxima de z$
p (2, tr + 1)	Cadena numérica. El primer elemento de cada par almacena la fecha en un código numérico. El segundo elemento señala la localización del dato dentro de z$.
nr	Número actual de datos ya introducidos.
m$	Cadena literal que almacena el número de días de cada mes.

TABLA TY1. **Nombre y utilización de las principales variables.**

El programa utiliza repetidamente 2 subrutinas: Una codifica una fecha y la otra busca el dato correspondiente cuando se le solicita. La primera subrutina ("FECHA") se encuentra entre las líneas 400 y 630. El programa pregunta al usuario:

- día del mes (cifra con 2 dígitos)
- nombre del mes (tres o más letras)
- año (cifra con los últimos dos dígitos como mínimo).

La subrutina comprueba la sintáxis de la fecha, incrementa el segundo elemento de (d) si el año es BISIESTO y almacena la fecha de forma numérica en la variable (da) usando un código simple. Así 19 diciembre 1982 es almacenado como 21219, por ejemplo (*).

La subrutina de "BUSQUEDA" se encuentra entre las líneas 7000 y 7270. Todos los datos introducidos en z$ se mantienen ordenados por fecha de entrada en el primer elemento de la cadena (p). Asimismo, para determinar si existe ya un dato almacenado en (da), es necesario comparar (da) con cada uno de los primeros elementos de la cadena (p). La subrutina siguiente ilustra un método simple para realizar esta búsqueda dentro de (p), donde (nr) es el número de datos del DIARIO.

```
10 FOR i = 1 TO nr
20 IF da = p (1, i) THEN PRINT "ENCONTRADO UN DATO": STOP
30 NEXT i
40 PRINT "NO HAY DATOS"
50 STOP
```

Esta técnica es necesariamente lenta porque es necesario comprobar cada valor de p (1, i). La subrutina "BUSQUEDA" utiliza un método más rápido llamado "DOBLE BUSQUEDA", que presupone que los datos se encuentran ordenados por la fecha de entrada. Los pasos que se siguen en este método son los siguientes:

1. Determina el elemento situado en el medio del recorrido de (p) para buscarlo.
2. Compara (da) con dicho elemento.
3. Si (da) es MENOR que ese elemento, desvía su atención hacia la mitad inferior de p, GO TO 1).
4. Si (da) es MAYOR que ese elemento, desvía su atención hacia la mitad superior de p, GO TO 1).

Supongamos, por ejemplo, que (da) es igual a p (1, 47) y el número de datos es 83. El proceso de búsqueda podría ser como sigue:

1.—Se compara da a p $\left(1, \frac{83}{2}\right)$, o sea a p (1,41)

2.—da es mayor que p (1,41) compara da con p $\left(1{,}41 + \frac{83 - 41}{2}\right)$ = p (1,62)

3.—da es menor que p (1,62) compara da con p $\left(1{,}41 + \frac{62 - 41}{2}\right)$ = p (1,51)

4.—da es menor que p (1,51) compara da con p $\left(1{,}41 + \frac{51 - 41}{2}\right)$ = p (1,46)

5.—da es mayor que p (1,46) compara da con p $\left(1{,}46 + \frac{51 - 46}{2}\right)$ = p (1,48)

6.—da es menor que p (1,48) compara da con p $\left(1{,}46 + \frac{48 - 46}{2}\right)$ = (1,47)

7.—da = p (1,47)

(*) NOTA DEL TRADUCTOR: 2, por 1982; 12 por el mes de DICIEMBRE; 19 por el día del mes.

Como puede observarse, la búsqueda se ha realizado en 7 etapas en lugar de las 47 que necesitábamos antes. El máximo número de etapas requerido para localizar un dato se obtiene mediante la ecuación

donde (nr) es el número de datos introducidos hasta ese momento y (m) el número entero más pequeño para el cual la inecuación es cierta.

COMO UTILIZAR EL PROGRAMA "DIARIO"

Lo primero que realiza el programa es una definición de las cadenas que va a necesitar. A continuación lee los datos de los meses y los días de cada mes en los DATA de la línea 200, asignándolos a M$(i) y d(i) mediante READ (líneas 230 y 260). Habiendo iniciado el programa de esta forma no debemos utilizar "RUN" para ejecutar el programa sino "GO TO 300", ya que de otra manera podríamos perder el contenido de las variables.

El programa pregunta la fecha actual, utilizando la subrutina "FECHA" y la compara con los datos de la cadena índice utilizando la subrutina "BUSQUEDA". Si existen datos para fechas anteriores que la fecha actual serán borrados (líneas 900 a 970).

El MENU de OPCIONES se encuentra en la línea 1010 y se obtiene por pantalla tal como se ilustra en la FIGURA FY1, pudiendo el usuario escoger entre ellas. Las opciones sobre VERIFICAR, IMPRIMIR o BORRAR una anotación son tan claras que no necesitan mayor explicación. Hacer uso de las rutinas "FECHA" y "BUSQUEDA" ya mencionadas para localizar el item requerido e imprimirlo o borrarlo si es necesario.

```
Diario    13 JULIO        1984
                Menu
Introducir anotaciones     I
Verificar anotaciones      V
Imprimir anotaciones       P
Borrar anotaciones         B
Parar el programa          H
```

FIGURA FY1. **El MENU de OPCIONES de "DIARIO"**

Un ejemplo de los resultados del programa se muestran en la Figura FY2. Los datos se refieren a una visita a la República Federal Alemana.

La opción INTRODUCIR un dato es un poco más compleja poco utilizada también las subrutinas básicas anteriores. Si exista ya un dato para la fecha en cuestión, le número datos se sitúa al final del dato ya existente pero separados ambos por una coma y un espacio. Si un dato no existe, el programa introduce el nuevo dato en su orden de fecha.

El programa, tal como está escrito, tiene capacidad para 100 datos (200 caracteres) y utiliza casi todo el espacio disponible en una máquina de 16K. Con una de 48K los valores de (tr) y (tz) -líneas 100 y 110- podrían aumentarse a 1700 y 34000 respectivamente.

```
Anotacion del 19 JULIO        1984:
13.30 Salida de Heathrow:
15.30 Vuelo BA 728 a Frankfurt


Anotacion del 20 JULIO        1984:
Visita a Karlsruhe


Anotacion del 22 JULIO        1984:
Visita a Munich


Anotacion del 27 JULIO        1984:
Visita a Wiesbaden

Anotacion del 29 JULIO        1984:
Visita a Saarbrucken

Anotacion del 30 JULIO        1984:
10.25 BA 725 Franckfurt a Heathrow
```

FIGURA FY2. **El itinerario de una visita a la República Federal Alemana.**

PROGRAMA PY1. **DIARIO**

```
100 LET tr=100
110 LET tz=2000
120 DIM p(2,tr+1)
130 DIM z$(1,tz)
140 LET nr=0
150 LET nr=0
```

```
200 DATA "ENERO","FEBRERO","MARZO","ABRIL","MAYO","JUNIO",
    "JULIO","AGOSTO","SEPTIEMBRE","OCTUBRE","NOVIEMBRE",
    "DICIEMBRE",31,28,31,30,31,30,31,31,30,31,30,31
210 DIM M$(12,10): DIM D(12)
220 FOR i=1 TO 12
230 READ m$(i): NEXT i
250 FOR i=1 TO 12: READ d(i): NEXT i
300 PRINT TAB 11; PAPER 6; INK 0;" Diario "; PAPER 7,,,TAB 4;
    PAPER 6;"Introduce la fecha de hoy"
305 GO SUB 8000
310 GO SUB 400
320 LET d$=STR$ fecha+" "+m$(i)+" "+STR$ y
330 GO SUB 7000
340 IF nr=0 THEN  GO TO 1000
350 IF e<0 THEN  LET e=-e-1: GO TO 900
360 LET e=e1
370 GO TO 900
410 PRINT ,,TAB 0; PAPER 6;"Introduce el dia, mes  y  ano  ";
     CHR$ 8;CHR$ 8;CHR$ 8;CHR$ 8; OVER 1;" "
420 PRINT ,,"Fecha : ",
430 INPUT fecha
440 PRINT fecha
450 PRINT ,,"Mes   : ",
460 INPUT n$: IF LEN n$<3 THEN  BEEP 1,12: GO TO 460
470 FOR i=1 TO 3
480 LET n$(i)=CHR$ (CODE n$(i)-32*(n$(i)>"`"))
490 NEXT i
500 FOR i=1 TO 12
510 IF n$(1 TO 3)=m$(i,1 TO 3) THEN  GO TO 540
520 NEXT i: BEEP 1,12: GO TO 460
540 PRINT m$(i)
550 PRINT ,,;"Ano  ";CHR$ 8;CHR$ 8;CHR$ 8;CHR$ 8; OVER 1;
    "      : ",
560 INPUT y: IF y<80 OR y>2000 THEN  BEEP 1,12: GO TO 560
570 IF y>=100 AND y<1980 THEN  BEEP 1,12: GO TO 560
580 LET y=y+1900*(y<1980)
590 PRINT y
595 PAUSE 100
600 LET d(2)=28+(y=4*INT (y/4) AND y<>2000)
610 IF fecha<1 OR fecha>d(i) THEN  BEEP 1,12: PRINT ,, PAPER 4;
    TAB 4;" Datos erroneos -repite ": PAUSE 100: GO TO 400
620 LET da=fecha+l*100+(y-1980)*10000
630 RETURN
900 LET long=p(2,e+1)
910 LET z$(1,1 TO p(2,nr+1)-long)=z$(1,p(2,e+1)+1 TO p(2,nr+1))
920 LET nr=nr-e
930 FOR i=1 TO nr: LET p(1,i)=p(1,e+i)
950 LET p(2,i)=p(2,e+1)-long
960 NEXT i
970 LET p(2,nr+1)=p(2,nr+e+1)-long
1000 CLS
1010 PRINT  PAPER 4;" Diario ";
1015 PRINT "  ";d$
1020 PRINT ,,TAB 12; PAPER 4;"Menu"
1021 PRINT ,, PAPER 6;"Introducir anotaciones   I "
1022 PRINT ,, PAPER 6;"Verificar anotaciones    V "
1023 PRINT ,, PAPER 6;"Imprimir anotaciones     P "
1024 PRINT ,, PAPER 6;"Borrar anotaciones       B "
1025 PRINT ,, PAPER 6;"Parar el programa        H "
1026 INPUT a$: IF CODE a$>96 THEN  LET a$=CHR$ (CODE a$-32)
1030 GO TO 1100+400*(a$="I")+900*(a$="V")+1400*(a$="P")+1900*
     (a$="B")+2400*(a$="H")
1100 PRINT ,, PAPER 4;"Pulsa una tecla para ir al menu "
1110 PAUSE 0
```

```
1120 GO TO 1000
1500 CLS : PRINT  PAPER 6;"  Introduccion de anotaciones  "
1510 GO SUB 400
1520 LET e$=STR$ fecha+" "+m$(i)+" "+STR$ y
1530 PRINT ,,"Cual es la anotacion ?"
1540 INPUT w$
1543 PRINT
1545 PRINT w$
1547 PRINT
1550 LET long=LEN w$
1600 IF nr<>0 THEN  GO TO 1700
1610 LET nr=nr+1
1620 LET p(1,1)=da
1630 LET p(2,1)=0: LET p(2,2)=long
1640 LET z$(1,1 TO long)=w$
1650 GO TO 1100
1700 GO SUB 7000
1710 IF e<0 THEN  LET e=-e: GO TO 1850
1720 LET e=eu
1750 LET nr=nr+1
1760 LET p(2,nr+1)=p(2,nr)+long
1770 FOR i=nr TO e+1 STEP -1
1780 LET p(1,i)=p(1,i-1): LET p(2,i)=p(2,i-1)+long
1790 NEXT i
1800 LET p(1,e)=da
1810 LET z$(1,1 TO p(2,nr+1))=z$(1,1 TO p(2,e))+w$+z$(1,p(2,e)+1
     TO p(2,nr+1))
1820 GO TO 1100
1850 LET z$(1,1 TO p(2,nr+1)+long+2)=z$(1,1 TO p(2,e+1))+":
     "+w$+z$(1,p(2,e+1)+1 TO p(2,nr+1))
1860 LET p(2,nr+1)=p(2,nr+1)+long+2
1890 NEXT i
1900 GO TO 1100
2000 CLS : PRINT TAB 2; PAPER 6;" Verificar una anotacion "
2010 GO SUB 400: GO SUB 7000
2020 LET e$=STR$ fecha+" "+m$(i)+" "+STR$ y
2030 IF e<0 THEN  LET e=-e: GO TO 2050
2040 PRINT ,, PAPER 4;"No hay nada anotado el ";e$: GO TO 1100
2050 PRINT ,, PAPER 4;"Anotacion del ";e$;":"
2060 PRINT ,,z$(1,p(2,e)+1 TO p(2,e+1))
2070 GO TO 1100
2500 CLS : PRINT TAB 3; PAPER 6;" Impresion de anotaciones  "
2510 GO SUB 400: GO SUB 7000
2520 LET e$=STR$ fecha+" "+m$(i)+" "+STR$ y
2530 IF e<0 THEN  LET e=-e: GO TO 2550
2540 LPRINT ,,"No hay nada anotado el ";e$: GO TO 1100
2550 LPRINT ,,"Anotacion del ";e$;":"
2560 LPRINT ,,z$(1,p(2,e)+1 TO p(2,e+1))
2570 GO TO 1100
3000 CLS : PRINT TAB 6; PAPER 6;" Borrar anotaciones "
3010 GO SUB 400: GO SUB 7000
3020 LET e$=STR$ fecha+" "+m$(i)+" "+STR$ y
3030 IF e<0 THEN  LET e=-e: GO TO 1100
3040 PRINT ,, PAPER 4;"No hay nada anotado el ";e$: GO TO 1100
3050 LET long=p(2,e+1)-p(2,e)
3060 LET z$(1,1 TO p(2,nr+1)-long)=z$(1,1 TO p(2,e))+
     z$(1,p(2,e+1)+1 TO p(2,nr+1))
3070 LET nr=nr-1
3080 FOR i=e TO nr
3090 LET p(1,i)=p(1,i+1)
3100 LET p(2,i)=p(2,i+1)-long
3110 NEXT i
3120 LET p(2,nr+1)=p(2,nr+2)-long
3130 GO TO 1100
```

```
3500 CLS : PRINT "Programa parado"
3510 STOP
6000 PRINT "Da",: INPUT da: PRINT da
6040 LET nr=10
6100 FOR i=1 TO 10
6110 PRINT i,p(1,i)
6120 NEXT i
7000 LET el=0: LET eu=nr+1
7010 IF eu=el+1 THEN  RETURN
7020 LET e=el+INT ((eu-el)/2)
7030 GO TO 7050+100*(da<p(1,e))+200*(da>p(1,e))
7050 LET e=-e: RETURN
7060 STOP
7160 LET eu=e
7170 GO TO 7010
7260 LET el=e
7270 GO TO 7010
8000 POKE USR "a"+1,0
8010 POKE USR "a",BIN 01111000
8020 FOR x=USR "a"+2 TO USR "a"+7: POKE x,0: NEXT x
8030 RETURN
```

3. CALENDARIO

Este programa puede ser utilizado para generar el calendario de cualquier mes de un año comprendido entre 1980 y el año 2000. El calendario para cada mes es visualizado en pantalla (o impreso) como un tablero con retícula de 7 x 5 con el día del mes situado aproximadamente en el centro de cada cuadrado y para todos los días hábiles del mes y año. Los días de la semana aparecen en la parte superior del tablero.

El programa fue escrito una vez terminado el anterior (DIARIO), por lo que utiliza algunas subrutinas y variables utilizadas allí. Por ejemplo, los nombres de cada mes y el número de días de cada uno se almacenan en (m$) y (d) en una sentencia DATA? en la cual también se almacenan los días de la semana como (d$).

Una vez comenzada la ejecución del programa se le pregunta al usuario si desea una copia de la hoja mensual de calendario que se obtiene como resultado final. Para ello se ha de introducir el mes y el año para los cuales se desea el calendario. Sólo son significativas las 3 primeras letras del nombre del mes y se pueden incorporar al programa tanto en letras mayúsculas como minúsculas. De la misma forma se pueden omitir los 2 primeros dígitos del año.

Por tanto, son aceptables las siguientes fechas:

Sept 82 SEPT 82 Sep 1982

Un calendario ejemplo, realizado para diciembre de 1982 se muestra en la figura FC1. Lo usual es que el calendario comience en el LUNES de la semana en que el mes empieza. Como el 1-12-82 fue MIERCOLES y el mes tiene 31 días hay 2 cuadrados vacíos tanto al comienzo como al final del mes.

DICIEMBRE 1982

Lun	Mar	Mie	Jue	Vie	Sab	Dom
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		

FIGURA FC1. El calendario para diciembre de 1982.

La situación puede complicarse cuando el primer día de un mes con 31 es SABADO o DOMINGO. Si la primera semana comenzara un lunes quedarían vacíos 5 ó 6 cuadrados antes del comienzo del mes y los 30 ó 31 días podrían requerir un febrero con 36 y 37 cuadrados.

Para resolver esta situación se almacenan 9 días (y no 7) en la cadena literal de los días de la semana (d$), siendo tanto los 2 primeros como los 2 últimos "SAB" y "DOM". Si un mes comienza un sábado o domingo el programa sitúa el primer cuadrado del tablero en ese día. El resultado se ilustra en la FIGURA FC2 para enero de 1983.

ENERO 1983

Sab	Dom	Lun	Mar	Mie	Jue	Vie
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

FIGURA FC2. **El calendario para enero de 1983**

PROGRAMA PC1. CALENDARIO

```
 5 BORDER 6: INK 0
 6 PAPER 5: CLS
10 PRINT TAB 4; FLASH 1;"Calendario para Spectrum"
20 PRINT ,,"Quieres el calendario para un mes o para un año?
   (m/a)"
30 INPUT z$: LET z$=CHR$ (CODE z$-32*(z$>"`")):
   IF z$<>"M" AND z$<>"A" THEN  BEEP 1,17: GO TO 30
```

```
 40 PRINT ,,"Quieres imprimirlo ? (s/n)"
 50 INPUT c$: LET c$=CHR$ (CODE c$-32*(c$)"`")):
    IF c$<>"S" AND c$<>"N" THEN  BEEP 1,17: GO TO 50
100 DATA "ENERO","FEBRERO","MARZO","ABRIL","MAYO","JUNIO",
    "JULIO","AGOSTO","SEPTIEMBRE","OCTUBRE","NOVIEMBRE",
    "DICIEMBRE","Sab","Dom","Lun","Mar","Mie","Jue","Vie","Sab",
    "Dom",31,28,31,30,31,30,31,31,30,31,30,31
110 DIM m$(12,10): DIM d$(9,3): DIM d(12)
120 FOR i=1 TO 12
130 READ m$(i)
140 NEXT i
150 FOR i=1 TO 9
160 READ d$(i)
170 NEXT i
180 FOR i=1 TO 12
190 READ d(i)
200 NEXT i
300 BORDER 4: INK 7: PAPER 1: CLS : BEEP 1,30
310 IF z$="A" THEN  GO TO 3000
500 PRINT TAB 6; FLASH 1;"Calendario de un mes.";
    FLASH 0,,," Introduce mes y año"
505 PRINT ,,
510 PRINT "Mes = ",
520 INPUT n$: IF LEN n$<3 THEN  BEEP 1,17: GO TO 520
530 FOR i=1 TO 3
540 LET n$(i)=CHR$ (CODE n$(i)-32*(n$(i))"`"))
550 NEXT i
560 FOR i=1 TO 12
570 IF n$(1 TO 3)=m$(i,1 TO 3) THEN  GO TO 600
580 NEXT i
590 BEEP 1,17: GO TO 510
600 PRINT  FLASH 1;m$(i); FLASH 0
605 PRINT ,,
610 PRINT "Ano =",
620 INPUT y: IF y<80 OR y>2000 THEN  BEEP 1,17: GO TO 620
640 LET y=y+1900*(y<1980)
650 PRINT  FLASH 1;y; FLASH 0
660 PAUSE 150
670 GO SUB 700
680 STOP
700 LET j=0
710 IF i=1 THEN  GO TO 800
720 LET d(2)=28+(y=4*INT (y/4) AND y<>2000)
730 FOR l=1 TO i-1
740 LET j=j+d(l)
750 NEXT l
800 LET j=j+4+(y-1980)*365+INT ((y-1981)/4)
810 LET j=j+1-7*INT (j/7)
1000 PAPER 7: INK 1: BORDER 6: CLS
1010 PRINT TAB 8;m$(i);TAB 19;y
1100 LET b=-2+4*(j-2+(3-j)*(j<3))
1110 LET a=5
1120 FOR l=1 TO d(i)
1130 PRINT AT a-1,b+1; INK 2;l; INK 1
1140 LET b=b+4
1150 IF b>26 THEN  LET b=2: LET a=a+4
1160 NEXT l
1500 LET j=j-(j-3)*(j>3)
1510 LET a=2
1520 FOR l=j TO j+6
1530 PRINT AT 1,a;d$(l)
```

```
1540 LET a=a+4
1550 NEXT l
2000 FOR a=15 TO 239 STEP 32
2010 PLOT a,0
2020 DRAW 0,167
2030 NEXT a
2050 FOR a=0 TO 167 STEP 32
2060 PLOT 15,a
2070 DRAW 224,0
2080 NEXT a
2090 IF c$="S" THEN  COPY
2095 IF INKEY$="" THEN  GO TO 2095
2100 RETURN
3000 PRINT TAB 6; FLASH 1;"Calendario para un año"; FLASH 0,,,"Int
roduce el ano"
3005 PRINT ,,
3010 PRINT "Ano =";
3020 INPUT y: IF y<80 OR y>2000 THEN  BEEP 1,17: GO TO 3020
3030 IF y>100 AND y<1980 THEN  BEEP 1,17: GO TO 3020
3040 LET y=y+1900*(y<1980)
3050 PRINT y
3060 PAUSE 150
3100 IF c$="S" THEN  LPRINT TAB 7;"Calendario para ";y:
     LPRINT : LPRINT
3110 FOR i=1 TO 12
3120 GO SUB 700
3130 IF c$="S" THEN  LPRINT : LPRINT
3140 NEXT i
```

4. AHORCADO

Esta es una versión del conocido juego, consistente en adivinar una palabra letra a letra. El programa comienza con un vocabulario muy limitado, pero aprende muy rápidamente. El usuario puede elegir entre ENSEÑARLE al programa las nuevas palabras bien mediante su introducción explícita o bien jugando contra la máquina. En este último caso el programa "AHORCADO" añade palabras aún desconocidas a su vocabulario.

La parte del programa comprendida entre las líneas 10 y 95 incluye varios parámetros que determinan las dimensiones relativas y la localización del AHORCADO. La figura del AHORCADO se dibuja mediante llamadas a las subrutinas de las líneas 8000, dependiendo la parte dibujada del valor obtenido en la subrutina correspondiente.

El programa toma su vocabulario inicial de los DATOS incluidos en la cadena z$. La longitud y número de palabras incluidos en los DATOS están en función del tamaño de (z$), dependiendo sólo que estén escritas en letras mayúsculas y que la última palabra sea "FIN".

El MENU de opciones se encuentra en la línea 1000, y aparece tal y como se ilustra en la Figura FJ1.

```
                    AHORCADO

Aumentar vocabulario                    I
ZX Spectrum adivina      palabraG
Tu adivinas una palabra                 Y
Grabar el      vocabulario              S
Cargar el vocabulario                   L
Detener      el programa                H
```

FIGURA FJ1. MENU de opciones de AHORCADO

PROGRAMA PJ1. AHORCADO

```
10 LET p1=0: LET p2=24
15 LET p3=32: LET p4=16
20 LET p5=p1: LET p6=p2+p4
25 LET p7=128
30 LET p8=p6+p7
35 LET p9=72
40 LET p10=p5+p9
45 LET p11=24
50 LET p12=8: LET p13=p8-p11-p12
55 LET p14=p13-p12: LET p15=32
60 LET p16=p13-2*p12
65 LET p17=24: LET p18=8
70 LET p19=p14-p15
75 LET p20=24: LET p21=40
```

```
 200 LET lz=3000
 210 DIM z$(lz+2)
 220 LET ez=0
 230 LET nz=0
 240 DATA "BYTE","CABEZON","SPECTRUM","PROGRAMA","COMPUTADOR",
     "HEXADECIMAL","FIN"
 250 READ x$
 260 LET z$( TO ez+LEN x$+1)=z$( TO ez)+CHR$ LEN x$+x$
 270 LET ez=ez+LEN x$+1
 280 LET nz=nz+1
 290 IF x$<>"FIN" THEN  GO TO 250
1000 CLS : PRINT ,,TAB 11;"AHORCADO"
1010 PRINT ,,,,"Aumentar vocabulario";TAB 31;"IZX Spectrum adivina
     palabra";TAB 31;"GTu adivinas una palabra";TAB 31;"YGrabar el
     vocabulario";TAB 31;"SCargar el vocabulario";TAB 31;"LDetener
     el programa";TAB 31;"H"
1020 INPUT y$: LET y$=CHR$ (CODE y$-32*(y$>"£"))
1030 GO TO 1100+400*(y$="I")+900*(y$="G")+1900*(y$="Y")+1400*
     (y$="S")+2400*(y$="L")+2900*(y$="H")
1100 INPUT "Pulsa ""ENTER"" para continuar";y$
1110 GO TO 1000
1500 PRINT ,,TAB 6;"Aumentar vocabulario"
1520 PRINT
1530 LET pw=1
1540 PRINT z$(pw+1 TO CODE z$(pw)+pw);", ";
1550 LET pw=pw+1+CODE z$(pw)
1560 IF pw<ez THEN  GO TO 1540
1570 PRINT
1600 INPUT "Introduce una palabra. ENTER    para terminar";x$
1610 IF x$="" THEN  GO TO 1000
1620 IF ez+LEN x$>=lz THEN  PRINT :
     PRINT ,,"No has dejado espacio": GO TO 1100
1630 FOR i=1 TO LEN x$
1640 LET x$(i)=CHR$ (CODE x$(i)-32*(x$(i)>"£"))
1650 NEXT i
1660 LET z$( TO ez+1+LEN x$)=z$( TO ez)+CHR$ LEN x$+x$
1670 LET ez=ez+1+LEN x$
1680 LET nz=nz+1
1690 GO TO 1600
2000 CLS : PRINT TAB 0;"ZX Spectrum adivina una palabra";
2010 INPUT "De cuantas letras ? ";nl
2020 IF nl<0 THEN  BEEP .2,24: GO TO 2010
2030 PRINT AT 5,16;nl;" letras"
2040 PRINT AT 20,(32-nl)/2;
2050 FOR i=1 TO nl
2060 PRINT "-";
2070 NEXT i
2080 PRINT
2090 LET s=0
2100 LET sub=7900
2110 DIM m(26)
2120 LET n=1+INT (26*RND)
2130 IF m(n)=1 THEN  GO TO 2120
2140 LET m(n)=1
2150 LET x$=CHR$ (64+n)
2170 PRINT AT 21,4;"La palabra contiene ";x$;" ?"
2180 INPUT y$: IF LEN y$<1 THEN  BEEP .2,24: GO TO 2180
2190 IF y$(1)="S" OR y$(1)="s" THEN  GO TO 2400
2200 LET sub=sub+100
2210 GO SUB sub
2220 IF sub<8900 THEN  GO TO 2120
2230 PRINT AT 7,16; FLASH 1;"Has ganado"
2240 PAUSE 300
2250 INPUT "Cual era la palabra ? ";x$
```

```
2260 IF LEN x$<>nl THEN  BEEP .2,24: GO TO 2250
2270 LET pw=1
2275 FOR i=1 TO nl: LET x$(i)=CHR$ (CODE x$(i)-32*(x$(i)>"£")):
     NEXT i
2280 IF x$=z$(pw+1 TO pw+CODE z$(pw)) THEN  GO TO 1100
2290 LET pw=pw+CODE z$(pw)+1
2300 IF pw<ez THEN  GO TO 2280
2310 IF ez+nl+1>=lz THEN  GO TO 1100
2320 LET z$( TO ez+nl+1)=z$( TO ez)+CHR$ nl+x$
2330 LET ez=ez+nl+1
2340 LET nz=nz+1: GO TO 1100
2400 INPUT "Introduce num. de veces ";n
2410 IF n<0 OR n>nl THEN  BEEP .2,24: GO TO 2400
2420 LET m=0
2430 INPUT "Introduce posicion letra ";i: IF i<1 OR i>nl THEN
     BEEP .2,24: GO TO 2430
2440 PRINT AT 20,(32-nl)/2+i-1;x$
2450 LET m=m+1: IF m<n THEN  GO TO 2430
2460 LET s=s+1: IF s<nl THEN  GO TO 2120
2470 PRINT AT 7,16; FLASH 1;"Mala suerte ! ": PAUSE 300:
     GO TO 1100
2500 PRINT ,,TAB 8;"Grabar vocabulario"
2510 INPUT "Introduce nombre ";y$
2520 IF LEN y$>10 THEN  BEEP .1,24: GO TO 2510
2530 LET z$(lz+1)=CHR$ (ez-256*INT (ez/256))
2540 LET z$(lz+2)=CHR$ (INT (ez/256))
2550 SAVE y$ DATA z$()
2560 INPUT "Rebobina la cinta y pulsa ENTER y PLAY";x$
2565 PRINT AT 11,8; FLASH 1;"    VERIFICANDO    "
2570 VERIFY y$ DATA z$()
2580 GO TO 1100
3000 LET n=1+INT (nz*RND)
3010 CLS : PRINT ,,TAB 12;"AHORCADO"
3020 LET nw=1: LET pw=1
3030 IF nw=n THEN  GO TO 3100
3040 LET nw=nw+1: LET pw=pw+CODE z$(pw)+1
3050 GO TO 3030
3100 LET nl=CODE z$(pw)
3110 PRINT AT 5,16;nl;" letras"
3120 PRINT AT 20,(32-nl)/2;
3130 FOR i=1 TO nl
3140 PRINT "-";
3150 NEXT i
3160 PRINT
3170 LET x$=z$(pw+1 TO pw+CODE z$(pw))
3180 DIM l(LEN x$)
3190 LET sub=7900
3200 INPUT "Introduce una letra ";y$
3205 LET y$=CHR$ (CODE y$-32*(y$>"£"))
3210 LET in=0: LET s=0
3220 FOR i=1 TO LEN x$
3230 IF l(i)=0 AND x$(i)=y$(1) THEN  LET l(i)=1:
     PRINT AT 20,(32-nl)/2+i-1;y$: LET in=1
3240 LET s=s+l(i)
3250 NEXT i
3260 IF in=0 THEN  LET sub=sub+100: GO SUB sub
3270 IF s=LEN x$ THEN  PRINT AT 7,16; FLASH 1;"Has acertado":
     PRINT  FLASH 1;AT 9,16;89-sub/100;" puntos":
     PRINT AT 20,(32-nl)/2; FLASH 1;x$: GO TO 1100
3280 IF sub<8900 THEN  GO TO 3200
3290 PRINT AT 7,16; FLASH 1;"Mala suerte !"; FLASH 0;AT 9,14;
     FLASH 1;"la palabra era"
```

```
3300 PAUSE 300
3310 PRINT AT 7,16;"              ";AT 9,14;"
3330 PRINT AT 20,(32-nl)/2; FLASH 1;x$
3340 PAUSE 300
3350 PRINT AT 20,(32-nl)/2;x$
3360 GO TO 1100
3500 PRINT ,,TAB 8;"Cargar el vocabulario"
3510 PRINT "Introduce el nombre"
3520 INPUT y$: IF LEN y$>10 THEN  BEEP .2,24: GO TO 3520
3530 LOAD y$ DATA z$()
3540 LET ez=CODE z$(lz+1)+256*CODE z$(lz+2)
3550 LET pw=1
3560 LET nw=0
3570 LET pw=pw+1+CODE z$(pw)
3580 IF pw<ez THEN  LET nw=nw+1: GO TO 3570
3590 GO TO 1100
4000 PRINT ,,"Programa detenido"
4010 STOP
7999 STOP
8000 PLOT p1,p2
8010 DRAW p3,0
8020 DRAW 0,p4
8030 DRAW -p3,0
8040 DRAW 0,-p4
8050 RETURN
8100 PLOT p5,p6
8110 DRAW 0,p7
8120 RETURN
8200 PLOT p5,p8
8210 DRAW p9,0
8220 RETURN
8300 PLOT p10,p8
8310 DRAW 0,-p11
8320 RETURN
8400 CIRCLE p10,p13,p12
8410 RETURN
8500 PLOT p10,p14
8510 DRAW 0,-p15
8520 RETURN
8600 PLOT p10,p16
8610 DRAW -p17,-p18
8620 RETURN
8700 PLOT p10,p16
8710 DRAW p17,-p18
8720 RETURN
8800 PLOT p10,p19
8810 DRAW -p20,-p21
8820 RETURN
8900 PLOT p10,p19
8910 DRAW p20,-p21
8920 RETURN
```

5. FUTBOL

Este juego imita los juegos propios de TV y está preparado para 1 ó 2 jugadores. El ganador es la persona que primero logra 15 goles.

El programa es bastante fácil, pero utiliza algunas técnicas que vale la pena resaltar, incluyendo un método simple de almacenamiento de GRAFICOS DEFINIDOS POR EL USUARIO; la utilización de los comandos OVER e INVERSE para obtener en pantalla las características requeridas, y del comando IN para interrogar al teclado.

La construcción y almacenamiento de caracteres gráficos se describe en el capítulo 14 del Manual de Programación Basic para el ZX SPECTRUM.

Se reservan 168 bytes en la parte alta del RAM para 21 caracteres, teniendo cada uno de ellos 8 bytes. La forma de cada uno de los caracteres está determinada por los mismos de cada uno de los bytes y pueden ser alterados por el usuario utilizando el comando POKE USR.

El programa FUTBOL utiliza 4 caracteres especiales **a)** el balón (letra B) **b)** el balón en caracteres inversos (letra C), **c)** los dos porteros (letra R) y **d)** los dos porteros en caracteres inversos (letra S). Los mismos que determinan la forma de estos caracteres se hallan inicialmente en sentencias DATA en las líneas 200 a 230 y son transferidas al área de Gráficos mediante las líneas de 300 a 460.

Consideremos el primer bucle entre las líneas 300 a 340. La línea 300 almacena el valor de la variable del sistema UDG, situando el principio del área de gráficos en la variable (u).

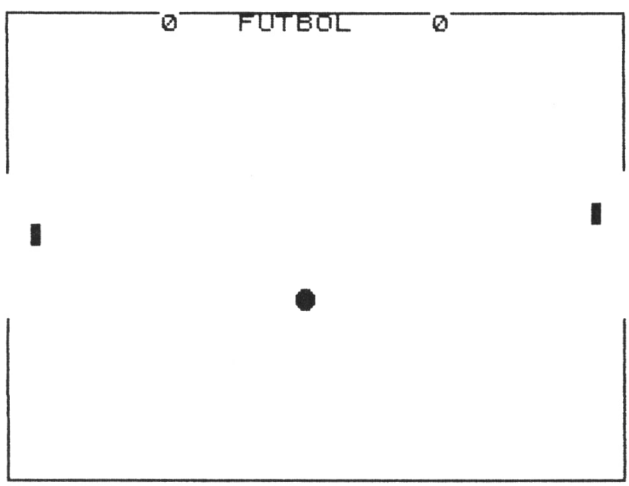


FIGURA FF1. Un partido de FUTBOL

El primer carácter que se almacena es el balón, que sustituye a la letra B. La B es la segunda letra del alfabeto, y así los valores leídos en las sentencias DATA son introducidos en direcciones POKE sucesivamente y en el segundo grupo de 8 bytes en el área de gráficos.

El siguiente bucle, líneas 350 a 380 realiza la misma operación con los siguientes 8 valores, situándolos en las localizaciones gráficas para la letra C, y de la misma forma llevamos a cabo los otros 2 cambios.

La línea 1000 dirige el balón alrededor del área de juego. La sentencia PRINT incluye OVER 1, así que, cuando la imagen se sitúa en una nueva localización PRINT el balón se oscurece, pero no elimina la imagen que ya estaba allí. El efecto se puede ver mucho mejor cuando el balón golpea una pared. Aquellos pixels donde no coinciden balón y pared toman el color PAPER, y donde coinciden se colorean con INK.

La línea 2000 realiza la función complementaria de borrar el balón en la posición precedente. Esto lo realiza medianta la impresión de una imagen inversa del balón (p.e.: una imagen con sólo los pixels que no están en el balón eliminado). Utilizamos OVER 1 una vez más, de tal forma que la nueva imagen se añade a la imagen del balón hasta formar un carácter completamente coloreado EXCEPTO EN LOS PIXELS DONDE EL BALON FUE DIBUJADO SOBRE (OVER) LA OTRA IMAGEN. Las características de INVERSE se utilizan aquí también, de forma que se borra el balón aunque alguna imagen fundamental reaparezca.

Una técnica idéntica es utilizada para mover los porteros. Los porteros son controlados desde el teclado por las 2ª y 4ª filas de comandos. Cualquiera de las teclas de la parte izquierda de la 2ª fila mueve el portero izquierdo hacia arriba. Igualmente la parte izquierda de la última fila le mueve hacia abajo y la parte derecha de las 2ª y 4ª filas controla el portero derecho.

El comando IN se utiliza para preguntar sobre los comandos tal como se describen en el capítulo 23 del Manual de Programación Basic para el ZX SPECTRUM.

Esto permite a ambos jugadores presionar sus teclas al mismo tiempo.

No sería posible realizar esto mediante INKEY$ puesto que este comando sólo permite presionar una tecla a la vez y, si se presiona más de una tecla a la vez, es como si no se presionara ninguna de ellas.

PROGRAMA PF1. FUTBOL

```
 10 BORDER 4: PAPER 4: INK 7: CLS
 20 PRINT TAB 12; FLASH 1;"FUTBOL"; FLASH 0
100 DRAW 255,0: DRAW 0,60: PLOT 255,116: DRAW 0,59:
    DRAW -255,0: DRAW 0,-60: PLOT 0,60: DRAW 0,-60
110 LET t=.0005: LET n=36
120 LET a=0: LET b=0
```

```
 200 DATA 60,126,255,255,255,255,126,60
 210 DATA 195,129,0,0,0,0,129,195
 220 DATA 60,60,60,60,60,60,60,60
 230 DATA 195,195,195,195,195,195,195,195
 300 LET u=PEEK 23675+256*PEEK 23676
 310 FOR i=0 TO 7
 320 READ j
 330 POKE u+8+i,j
 340 NEXT i
 350 FOR i=0 TO 7
 360 READ j
 370 POKE u+16+i,j
 380 NEXT i
 390 FOR i=0 TO 7
 400 READ j
 410 POKE u+136+i,j
 420 NEXT i
 430 FOR i=0 TO 7
 440 READ j
 450 POKE u+144+i,j
 460 NEXT i
 500 LET u=2*INT (2*RND)-1: LET v=2*INT (2*RND)-1
 510 LET c=15+(v+1)/2: LET r=INT (22*RND)
 520 PRINT  FLASH 1;AT 0,8;a;AT 0,22;b
 600 LET p=10: LET q=10: PRINT  OVER 1;AT p,1;"▮":
     PRINT  OVER 1;AT q,30;;"▮"
 610 PAUSE 50
1000 PRINT  OVER 1;AT r,c;"●"
1010 LET or=r: LET oc=c
1020 IF c=1 AND r=p THEN  LET c=2*(v<0): LET r=r+u:
     LET v=-v: GO TO 1100
1030 IF c=30 AND r=q THEN  LET c=29+2*(v<0): LET r=r+u:
     LET v=-v: GO TO 1100
1040 LET r=r+u: LET c=c+v
1050 IF r<0 OR r>21 THEN  BEEP t,n: LET r=21*(r>21): LET u=-u
1060 IF (r<7 OR r>13) AND (c<0 OR c>31) THEN  BEEP t+.5,n:
     LET c=31*(c>31): LET v=-v
1100 IF IN 57342<255 THEN  PRINT  OVER 1; INVERSE 1;AT q,30;"||":
     LET q=q-(q<>0): PRINT  OVER 1;AT q,30;"▮"
1110 IF IN 32766<255 THEN  PRINT  OVER 1; INVERSE 1;AT q,30;"||":
     LET q=q+(q<>21): PRINT  OVER 1;AT q,30;"▮"
1120 IF IN 64510<255 THEN  PRINT  OVER 1; INVERSE 1;AT p,1;"||":
     LET p=p-(p<>0): PRINT  OVER 1;AT p,1;"▮"
1130 IF IN 65278<255 THEN  PRINT  OVER 1; INVERSE 1;AT p,1;"||":
     LET p=p+(p<>21): PRINT  OVER 1;AT p,1;"▮"
2000 PRINT  OVER 1; INVERSE 1;AT or,oc;"[]"
2010 IF c>=0 AND c<=31 THEN  GO TO 1000
3000 PRINT  OVER 1; INVERSE 1;AT p,1;"||"
3010 PRINT  OVER 1; INVERSE 1;AT q,30;"||"
3020 LET a=a+(c>31): LET b=b+(c<0)
3030 FOR i=0 TO 8
3040 PRINT  FLASH 1; INK i;AT 0,8;a;AT 0,22;b
3050 BEEP .1,i
3060 NEXT i
3080 IF a<15 AND b<15 THEN  GO TO 500
3090 PAUSE 50: RUN
```

NOTA: Los caracteres graficos de las lineas 1000 y 2000 corresponden con B y C respectivamente

6. MUSICA

He aquí un programa que le ayudará a usted a escribir música utilizando el alatavoz incorporado al SPECTRUM. El programa puede ser utilizado para escribir y modificar una determinada partitura (representando los datos en la pantalla o mediante la impresora); cambiar la melodía de CLAVE (do, sol, fa) y cambiarla también de "RITMO" (presto, adagio), Por supuesto que el "TONO" (do, re, mi) puede también modificarse, y además puede ser grabado separadamente en cassette, verificado y vuelto a cargar en una fecha posterior.

El programa hace un uso extensivo del comando "BEEP", el cual se explica con algún detalle en el capítulo 19 del Manual de Programación Basic para el ZX SPECTRUM. Este comando se utiliza en la forma:

BEEP duración, tono

donde la duración en el TIEMPO EN SEGUNDOS y el tono indica los SEMITONOS en relación con la nota DO CENTRAL (*). Cuando el usuario está escribiendo una pieza, lo que está realmente construyendo es una secuencia de duraciones de una forma satisfactoria, escogiendo parejas de valores que el programa almacena en una cadena de 2 dimensiones.

Cuando el programa se ejecuta (mediante "RUN"), accede a una subrutina de inicialización que determina los datos de TONO y RITMO (número de notas dentro de cada compás) y utiliza una cadena para almacenar dichas notas.

Los valores normales de una cadena tipo pueden ser: una longitud de 500 para almacenar la pieza musical, entre 300 y 400 sonidos por minutos y el dato relativo a la posición del DO CENTRAL (que se selecciona mediante Ø). El TONO y el RITMO se almacenan en los dos primeros elementos de la cadena para mayor tranquilidad cuando se GRABA la pieza en cinta.

El usuario controla el programa mediante el MENU ilustrado en la figura FS1, el cual se genera en la línea 200 del programa. Tecleando las letras de la columna de la derecha del MENU el programa se dirige a la subrutina apropiada, tal como le indica la línea 220.

(*) (NOTA DEL TRADUCTOR)

La notación musical germánica (D) y anglosajona (U.K., USA) es distinta a la latina (L), existiendo entre ambas las siguientes equivalencias (en líneas generales):

LATINA	(D + UK)	LATINA	(D + UK)
DO (UT)	C	FA	F
RE	D	SOL	G
MI	E	LA	A
		SI	B

DO CENTRAL es, en el clave de Sol en 2ª línea, la nota situada en la 1ª línea ADICIONAL DESCENDENTE del pentagrama.

```
MUSICA en el Spectrum

        MENU:

escribir            W
pantalla            D
impresora           LP
interpretar         P
nuevos valores      B
cargar              L
grabar              S
verificar           V
parar               H
```

FIGURA FS1. **MENU para MUSICA-SPECTRUM** **(TECLEAR W)**

ESCRIBIR UNA PIEZA MUSICAL

Las notas se han de introducir en el programa mediante grupos de 3 cifras. Primeramente el programa pregunta por el NUMERO (ORDINAL) de la nota de comienzo -normalmente 1-, para continuar con el RITMO (número de Sonidos por minuto) y terminar con el TONO (en semitono sobre el DO CENTRAL). Por ejemplo, una nota con 4 tiempos y 2 semitonos sobre el DO CENTRAL se debe introducir mediante un 4 seguido por un 2.

Si se introduce un número erróneo se debe teclear -999 para volver a la nota anterior. Para volver al MENU se teclea 999.

SALIDAS POR PANTALLA E IMPRESORA (Teclear D o LP)

Esta opción hace aparecer en la pantalla o por impresora la secuencia de notas actual.

NUEVOS VALORES (COMPAS Y CLAVE MUSICAL) (Teclear B)

Cuando se elija esta opción el programa preguntará por los nuevos valores del número de sonidos por minuto y el tono (relativo al DO CENTRAL). Dichos valores quedarán incorporados a la melodía actual.

INTERPRETACION DE LA MELODIA (Teclea P)

Esta subrutina interpreta la melodía actual. Si se pulsa una tecla mientras la melodía está sonando el programa se para e indica por pantalla el número de la última nota interpretada. Esta facultad es muy útil para eliminar de forma inmediata los errores que se adviertan en la melodía.

Su TABLA TS1 muestra las notas de la célebre melodía "TRES RATONES CIEGOS". Se requieren 51 notas como mínimo para interpretarla y sugerimos como más indicado que se interprete con 500 notas por minuto en la clave número 12.

Notas de la cancion

Nota no.	Duracion	Tono
1	4	4
2	4	2
3	8	0
4	4	4
5	4	2
6	8	0
7	4	7
8	2	5
9	2	5
10	8	4
11	4	7
12	2	5
13	2	5
14	8	4
15	1.5	7
16	2.5	12

17	1	12
18	1	11
19	1	9
20	1	11
21	1	12
22	1	7
23	1	7
24	3	7
25	1.5	7
26	2.5	12
27	1	12
28	1	11
29	1	9
30	1	11
31	1	12
32	1	7
33	1	7
34	3	7
35	1.5	12
36	2.5	12
37	1	12
38	1	11
39	1	9
40	1	11
41	1	12
42	1	7
43	1	7
44	2	7
45	1	5
46	4	4
47	4	2
48	8	0
49	4	4
50	4	2
51	8	0

TABLA TS1. **Las notas de "TRES RATONES CIEGOS"**

Quizás la manera más evidente en que podría desarrollarse este programa es permitir al usuario introducir las letras convencionales (notación germánica), con sostenidos (#) y bemoles (b) cuando se está escribiendo la pieza musical.

Esto podría llevarse a cabo mediante el almacenamiento de números, símbolos, etc. en una cadena literal de referencia, comparando los caracteres introducidos por el usuario con la cadena de referencia y obteniendo así los números de las notas tal como las admite el ZX SPECTRUM. El resto del programa sería prácticamente idéntico a éste:

PROGRAMA PS1. **MUSICA**

```
 10 CLS : PRINT TAB 7;"MUSICA en el Spectrum",,;TAB 0;
    "Introduce:",,,;TAB 5;"  No. total de notas",;TAB 5;
    "  No.de sonidos por minuto",;TAB 5;"  Tecla del DO central"
 20 INPUT nn,nb,k
 30 LET nb=nb/60
 40 LET end=0
 50 DIM B(nn+2,2)
 60 LET b(1,1)=nn
 70 LET b(1,2)=nb
 80 LET b(2,1)=k
200 CLS : PRINT TAB 7;"MUSICA en el Spectrum",,,TAB 14;"MENU:"
    ,,,TAB 7;"escribir";TAB 24;"W",TAB 7;"pantalla";TAB 24;"D",
    TAB 7;"impresora";TAB 24;"LP",TAB 7;"interpretar";TAB 24;"P",
    TAB 7; "nuevos valores";TAB 24;"B",TAB 7;"cargar";TAB 24;"L",
    TAB 7;"grabar";TAB 24;"S",TAB 7;"verificar";TAB 24;"V",TAB 7;
    "parar";TAB 24;"H"
205 PRINT
206 PRINT
210 INPUT z$
220 GO TO 300+200*(z$="W")+700*(z$="B")+1200*(z$="D")+1700*
    (z$="L")+2200*(z$="S")+2700*(z$="P")+3200*(z$="V")+3700*
    (z$="H")+4200*(z$="LP")
300 PAUSE 200
310 GO TO 200
500 PRINT "  Escribe o compon una cancion",,,
   "no.en que deseas comenzar","a escribir"
510 INPUT j: IF j<1 OR j>nn THEN  GO TO 510
520 PRINT ,,"Introduce la duracion y el       tono de las notas."
    ,,,"Introduce:",,"999 pra terminar",
    "-999 para volver para atras"
540 PRINT ,,"  Nota no.     Duracion  Tono"
550 FOR i=j+2 TO nn+2
560 INPUT a: IF a=-999 THEN  LET i=i-1: GO TO 580
570 IF a=999 THEN  GO TO 900
580 INPUT b: IF b=-999 THEN  GO TO 560
590 LET b(i,1)=a/nb: LET b(i,2)=k+b
600 PRINT TAB 5;i-2;TAB 15;nb*b(i,1);TAB 27;b(i,2)-k
610 NEXT i
620 PRINT ,,"no queda espacio"
630 LET end=nn+2
640 GO TO 300
900 IF i>end THEN  LET end=i-1
910 GO TO 200
1000 PRINT ,,"        Cambiar ritmo y clave",,,
     "Los valores actuales son:","  Notas/minuto ",nb*60,
     "  Tecla",k,,,"Introduce nuevos valores"
1010 INPUT nc,l
1020 FOR i=3 TO end
1030 LET b(i,1)=b(i,1)*nb*60/nc
1040 LET b(i,2)=b(i,2)-k+l
1050 NEXT i
```

```
1060 LET nb=nc/60
1070 LET k=1
1080 GO TO 60
1500 PRINT ,,TAB 10;"   Notas de la cancion",,
     "  Nota no.     Duracion  Tono"
1510 FOR i=3 TO end
1520 PRINT TAB 5;i-2;TAB 15;nb*b(i,1);TAB 27;b(i,2)-k
1530 NEXT i
1540 PRINT ,,"Pulsa tecla para volver al menu"
1550 PAUSE 32768
1560 GO TO 200
2000 PRINT ,,"           Cargar cancion",,,"Nombre de la cancion"
2010 INPUT z$: IF LEN z$>10 THEN
     PRINT "Nombre demasiado largo": GO TO 2010
2020 LOAD z$ DATA b()
2030 LET nn=b(1,1)
2040 LET nb=b(1,2)
2050 LET k=b(2,1)
2060 LET end=b(2,2)
2070 PRINT ,,z$;" Bien cargado"
2080 GO TO 300
2500 PRINT ,,"           Grabar programa",,,"Nombre de la cancion"
2510 INPUT z$: IF LEN z$>10 THEN
     PRINT "Nombre demasiado largo": GO TO 2510
2520 LET b(2,2)=end
2530 SAVE z$ DATA b()
2540 PRINT ,,z$;" Grabado"
2550 GO TO 300
3000 PRINT ,,"           Tocar una cancion"
3010 PAUSE 100
3020 FOR i=3 TO end
3030 BEEP b(i,1),b(i,2)
3040 IF INKEY$<>"" THEN  GO TO 3100
3050 NEXT i
3060 GO TO 200
3100 PRINT ,,"Interrumpido en la nota ";i-2,,TAB 0;
     "Continuas? (s/n)"
3110 INPUT z$: IF z$<>"s" AND z$<>"n" THEN  GO TO 3110
3120 IF z$="s" THEN  GO TO 3030
3130 GO TO 300
3500 PRINT ,,"     Verificar la cancion",,,"Nombre de la cancion"
3510 INPUT z$: IF LEN z$>10 THEN
     PRINT "Nombre demasiado largo": GO TO 3510
3520 VERIFY z$ DATA b()
3530 PRINT ,,z$;" Verificado"
3540 GO TO 300
4000 PRINT ,,"Programa detenido"
4010 STOP
4500 LPRINT ,,"          Notas de la cancion",,,
     "  Nota no.     Duracion  Tono"
4510 FOR i=3 TO end
4520 LPRINT "     ";i-2;TAB 15;nb*b(i,1);TAB 27;b(i,2)-k
4530 NEXT i
4540 GO TO 300
```

7. REPRESENTACION DE DATOS

He escrito este programa, así como los dos siguientes (REGRESION E HISTOGRAMA) para demostrar que el ZX SPECTRUM puede ser utilizado también para aplicaciones serias (en este caso ANALISIS Y DATOS) en igual medida como lo puede ser para tareas frívolas, normalmente asociadas con un ordenador casero. Los 3 programas se han diseñado de la forma más similar posible y utilizan la misma estructura de variables; de esta manera los datos introducidos en uno de los programas pueden ser GRABADOS en cinta y posteriormente CARGADOS en otra distinta. Tomados en su conjunto los 3 programas forman el núcleo de un "PAQUETE" para el análisis estadístico de datos.

La primera etapa del análisis es "dibujar los datos" de una forma tal que resulte adecuada al objetivo deseado y proporcione además una impresión visual de su dispersión. La inspección visual es el método más fácil y directo de comprobación de datos y detección de valores erróneos, tanto por medición incorrecta o por error en la introducción de datos en el computador.

En este programa la entrada de datos se puede realizar mediante teclado o mediante programas de datos en cinta, y la salida de datos numéricos puede hacerse por pantalla o por impresora, pudiendo además dibujar los datos, rectificar los valores erróneos y grabarlos en cassette.

UN BOSQUEJO DEL PROGRAMA

Cuando se ejecuta, el programa ofrece al usuario la posibilidad de optar entre cargar los datos de cinta o mediante teclado. En este último caso el usuario debe introducir también el número de puntos que han de ser dibujados (n). El programa utiliza la cadena numérica x (2, n + 1) para almacenar los datos, introduciendo el valor de (n) en x (1, 1). Una vez hecho esto, el usuario debe introducir ordenadamente n pares de valores x e y.

Cuando los datos ya se han introducido (o se ha completado la carga de cinta) aparece el menú de opciones en la pantalla (línea 1000) tal como se ilustra en la Figura FD1 y, una vez elegida una opción salta a la subrutina seleccionada por el usuario (línea 1050). Sólo la subrutina para representación de datos requiere una explicación detallada, siendo autoexplicativos los demás conceptos.

```
Representacion de datos

                    MENU

Revisar datos                R
Listar datos                 L
Corregir un valor            C
Imprimir puntos              P
Grabar datos                 S
Verificar datos              V
Detener programa             H
```

FIGURA FD1. El menú de opciones de "representación de datos"

El computador dibuja los valores de x e y tanto mínimos como máximos, e indica al usuario que puede introducir los valores mínimos y máximos de ambas variables, las cuales se utilizarán para dibujar cada eje. Esto da al usuario la oportunidad de dibujar sólo la parte de los datos que él desea. También puede utilizar si desea números "redondos" para los extremos del dibujo.

El programa dibuja los datos tal como se ilustran en la Figura FD2, omitiendo los valores fuera del rango elegido. El número de puntos no dibujados aparece en la parte inferior izquierda del diagrama. Los ejes coordenados presentan 5 valores cada uno (lo que proporciona 4 intervalos). Es necesario que cada uno de estos valores se trunquen a sólo 4 caracteres (en razón del espacio disponible y la claridad). Como el programa ajusta la posición de impresión de los valores del eje x por la derecha es necesario tener cuidado de que las cifras no pasen al renglón siguiente, debido a lo cual se truncan.

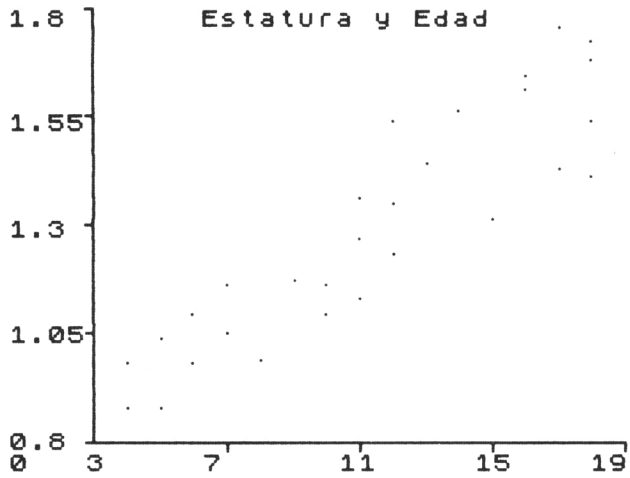


FIGURA FD2. **Una representación de la estatura-edad de 30 escolares**

El ejemplo que se muestra en la Figura FD2 se refiere a la estatura de 30 escolares (representada en ordenadas, eje y) puesta en relación con su edad (representado en abcisas, eje x). Los datos se han listado en la Tabla TD2.

La representación gráfica de la "nube de puntos" muestra un incremento uniforme de peso a medida que aumenta la edad del escolar, con tendencia a una mayor dispersión de valores conforme aumenta dicha edad.

TABLA TD1. **Los datos. La columna 1 es sólo un índice para la identificación numérica de cada escolar. La columna 2 es su edad (redondeada, en años) y la columna 3 es la estatura (en centímetros, expresada en metros) de cada uno de ellos.**

Indice	X	Y
1	7	1.05
2	12	1.54
3	7	1.16
4	10	1.16
5	11	1.13
6	4	0.88
7	14	1.56
8	10	1.09
9	6	1.09
10	12	1.23
11	16	1.61
12	17	1.75
13	13	1.44
14	5	1.04
15	6	0.98
16	15	1.31
17	6	1.09
18	16	1.64
19	4	0.98
20	9	1.17
21	18	1.68
22	5	0.88
23	8	0.99
24	18	1.54
25	11	1.27
26	18	1.41
27	11	1.36
28	12	1.35
29	17	1.43
30	18	1.72

PROGRAMA PD1. REPRESENTACION DE DATOS

```
  10 LET t=24
 500 PRINT ,,TAB 5;"Representacion de Datos",
     "Vas a introducir los datos desdeel cassette o desde el
     teclado  (C o T)?"
 510 INPUT z$
 520 IF CODE z$>96 THEN  LET z$=CHR$ (CODE z$-32)
 530 IF z$<>"T" AND z$<>"C" THEN  BEEP 1,12: GO TO 510
 540 GO TO 550+200*(z$="T")
 550 PRINT ,,"Introduce el nombre del programa"
 560 INPUT z$: IF LEN z$>10 THEN  BEEP 1,12: GO TO 560
 570 PRINT AT 21,0; FLASH 1;"Cargando el programa"
 580 LOAD z$ DATA x()
 590 PRINT AT 19,0;z$;" bien cargado"
 600 LET n=x(1,1)
 610 GO TO 1100
 750 PRINT ,,"Introduce el no. de puntos a    almacenar     "
 760 INPUT n: IF n<1 THEN  BEEP 1,12: GO TO 760
 770 DIM x(2,n+1)
 780 PRINT "Introduce ordenadamente los     valores para X e Y",,,
     "Indice";TAB 14;"X";TAB 26;"Y"
 790 FOR i=2 TO n+1
 795 PRINT i-1;TAB 14;
 800 INPUT x(1,i): PRINT x(1,i);TAB 26;
 810 INPUT x(2,i): PRINT x(2,i),
 820 NEXT i
 830 LET x(1,1)=n
1000 CLS
1010 PRINT TAB 0;"Representacion de datos"
1020 PRINT ,,,,TAB 14;"MENU"
1030 PRINT ,,"Revisar datos";TAB t;"R","Listar datos";TAB t;"L",
     "Corregir un valor";TAB t;"C","Imprimir puntos";TAB t;"P",
      "Grabar datos";TAB t;"S","Verificar datos";TAB t;"V",
     "Detener programa";TAB t;"H"
1040 INPUT z$: IF CODE z$>96 THEN  LET z$=CHR$ (CODE z$-32)
1050 GO TO 1100+400*(z$="R")+900*(z$="L")+1400*(z$="C")+1900*
     (z$="P")+3900*(z$="S")+4400*(z$="V")+4900*(z$="H")
1100 PRINT ,,"Pulsa tecla para volver al menu"
1110 PAUSE 0
1120 GO TO 1000
1500 PRINT ,,TAB 10;"Revisar datos",,,"Indice";TAB 14;"X";
     TAB 26;"Y"
1510 FOR i=2 TO n+1
1520 PRINT i-1;TAB 14;x(1,i);TAB 26;x(2,i)
1530 NEXT i
1540 GO TO 1100
2000 PRINT ,,TAB 11;"Listar datos",,,
2010 LPRINT ,,"Indice";TAB 14;"X";TAB 26;"Y"
2020 FOR i=2 TO n+1
2030 LPRINT i-1;TAB 14;x(1,i);TAB 26;x(2,i)
2040 NEXT i
2050 GO TO 1100
2500 PRINT ,,TAB 10;"Corregir datos"
2510 PRINT ,,"Introduce el indice del valor a cambiar"
2520 INPUT a: IF a<1 OR a>n THEN  BEEP 1,12: GO TO 2520
2530 PRINT ,,"Los valores actuales son :";TAB 8;x(1,a+1);
     TAB 20;x(2,a+1)
2540 PRINT ,,"Introduce ordenadamente los
     nuevos valores para X e Y"
2550 INPUT x(1,a+1),x(2,a+1)
2560 GO TO 1100
```

```
3000 PRINT ,,TAB 11;"Impresion de datos"
3010 LET xn=x(1,2)
3020 LET xx=x(1,2)
3030 LET yn=x(2,2)
3040 LET yx=x(2,2)
3050 FOR i=2 TO n+1
3060 IF xn>x(1,i) THEN  LET xn=x(1,i)
3070 IF xx<x(1,i) THEN  LET xx=x(1,i)
3080 IF yn>x(2,i) THEN  LET yn=x(2,i)
3090 IF yx<x(2,i) THEN  LET yx=x(2,i)
3100 NEXT i
3110 PRINT ,,"Menor valor de x = ",xn
3120 PRINT ,,"Mayor valor de x = ",xx
3130 PRINT ,,"Menor valor de y = ",yn
3140 PRINT ,,"Mayor valor de y = ",yx
3200 PRINT ,,"Introducir X min,X max,Y min,Ymax,  para imprimir"
3210 PRINT "X minima = ",: INPUT xn: PRINT xn
3220 PRINT "X maxima = ",: INPUT xx: PRINT xx
3230 PRINT "Y minima = ",: INPUT yn: PRINT yn
3240 PRINT "Y maxima = ",: INPUT yx: PRINT yx
3300 PRINT ,,"Introduce el titulo  "
3310 INPUT t$
3320 IF t$="" THEN  GO TO 3400
3330 PRINT ,,"Linea y columna donde se va a   imprimir el titulo"
3340 INPUT tl,tc
3350 IF tl<0 OR tl>21 OR tc<0 OR tc>31 THEN  BEEP 1,12: GO TO 3340
3400 PRINT ,,"Quieres una copia impresa        (S o N)?"
3410 INPUT c$: LET c$=CHR$  (CODE c$-32*(CODE c$>96))
3420 IF c$<>"S" AND c$<>"N" THEN  BEEP 1,12: GO TO 3410
4000 CLS
4010 LET x=35: LET y=11
4020 DRAW  INVERSE 1; OVER 1;x,y
4030 FOR i=1 TO 4
4040 DRAW 0,41
4050 DRAW -3,0
4060 DRAW 3,0
4070 NEXT i
4080 DRAW 0,-164
4090 DRAW -3,0
4100 DRAW 3,0
4110 DRAW -3,0
4120 DRAW 3,0
4130 FOR i=1 TO 4
4140 DRAW 55,0
4150 DRAW 0,-3
4160 DRAW 0,3
4170 NEXT i
4180 DRAW -228,0
4200 LET a=4
4210 FOR i=xn TO 1.01*xx STEP (xx-xn)/4
4220 LET z$=STR$ i
4230 IF LEN z$>4 THEN  LET z$=z$( TO 4)
4240 IF a=31 THEN  LET a=32-LEN z$
4250 PRINT AT 21,a;z$
4270 LET a=a+6+(a>9)
4280 NEXT i
4400 LET a=20
4410 FOR i=yn TO 1.01*yx STEP (yx-yn)/4
4420 LET z$=STR$ i+"   "
4430 PRINT AT a,0;z$( TO 4)
4440 LET a=a-5
4450 NEXT i
```

```
4500 LET o=0
4510 FOR i=2 TO n+1
4520 IF x(1,i)<xn OR x(1,i)>xx OR x(2,i)<yn OR x(2,i)>yx THEN
     LET o=o+1: GO TO 4600
4530 LET x=35+220*(x(1,i)-xn)/(xx-xn)
4540 LET y=11+164*(x(2,i)-yn)/(yx-yn)
4550 PLOT x,y
4600 NEXT i
4610 PRINT AT 21,0;o
4700 IF t$="" THEN  GO TO 4800
4710 PRINT AT tl,tc;t$
4800 IF c$="S" THEN  COPY
4900 PAUSE 0
4910 GO TO 1000
5000 PRINT ,,TAB 11;"Grabar datos",,,
     "Introduce nombre del programa"
5010 INPUT z$: IF LEN z$>10 THEN  BEEP 1,12: GO TO 5010
5020 PRINT AT 21,0; FLASH 1;"Grabando"
5030 SAVE z$ DATA x()
5040 PRINT AT 21,0;z$;" bien grabado"
5050 GO TO 1100
5500 PRINT ,,TAB 10;"Verificar datos",,,
     "Intrduce nombre del programa"
5510 INPUT z$: IF LEN z$>10 THEN  BEEP 1,12: GO TO 5510
5520 PRINT AT 21,0; FLASH 1;"Verificando"
5530 VERIFY z$ DATA x()
5540 PRINT AT 19,0;z$;" verificado"
5550 GO TO 1100
6000 PRINT ,,"programa detenido"
6010 STOP
```

8. REGRESION

Este es el segundo de los programas de análisis estadístico de datos, siendo los otros dos "REPRESENTACION DE DATOS" (programa anterior) e "HISTOGRAMA" (programa siguiente). El programa ajusta una recta a las parejas de valores x e y utilizando la técnica matemática de los mínimos CUADRADOS, obteniendo los coeficientes de la recta ajustada, los errores estimados de dichos coeficientes y el error estimado de la recta ajustada. También calcula los valores ajustados, es decir: los valores de y obtenidos por sustitución de los valores dados de x en la ecuación de la recta ajustada, y los residuos las diferencias entre los valores observados de la variable y los valores ajustados). Tanto los valores ajustados como los residuos pueden ser grabados en cinta en el mismo formato de los datos originales, de tal forma que puedan ser analizados utilizando los programas "REPRESENTACION DE DATOS" e "HISTOGRAMA".

Una vez ejecutado el programa se pregunta al usuario si los datos van a cargarse de cassette o a introducirse por teclado. Después aparece en pantalla el menú de opciones que se muestra en la Figura FR1. Las opciones son las mismas que en el programa anterior excepto la opción "REGRESION LINEAL" (CALCULAR LA REGRESION). Así el usuario puede visualizar los datos en pantalla o por impresora, puede corregir los valores erróneos y GRABAR y VERIFICAR los datos.

```
                    REGRESION

                       Menu

Revisar datos                    R
Listar datos                     L
Corregir un valor                C
Calcular la regresion            F
Grabar datos                     S
Verificar los datos              V
Detener el programa              H
```

FIGURA FR1. El MENU de opciones de REGRESION

Si el usuario selecciona la (opción "REGRESION LINEAL" el programa calcula **a)** media de los valores de y (yb). **b)** la suma de los valores de x (sxx) y de y (syy) elevados ambos al cuadrado y **c)** la suma de los productos de las variables x e y (sxy). La forma más simple de expresar las ecuaciones para obtener los conceptos anteriores es:

$$xb = \sum_{i=1}^{n} xi / n \qquad yb = \sum_{i=1}^{n} yi / n$$

$$s_{xx} = \sum_{i=1}^{n} (xi - xb)^2 = \sum_{i=1}^{n} (xi - xb) * (xi - xb)$$

$$s_{yy} = \sum_{i=1}^{n} (yi - yb)^2 = \sum_{i=1}^{n} (yi - yb) * (yi - yb)$$

$$s_{xy} = \sum_{i=1}^{n} (xi - xb) * (yi - yb)$$

donde n es el número de parejas de valores del ejemplo y Σ significa "SUMA DE VALORES DE...".

Como puede notarse, los valores de sxx, syy y sxy sólo pueden determinarse después de calcular xb e yb. Asimismo, para aplicar estas ecuaciones tal y como las hemos expresado antes del programa tendría que realizar 2 pasadas sobre los datos de entrada. Durante la primera pasada obtendríamos la suma de los valores de x y de y, con lo cual, al final de la pasada los dividiríamos por n y obtendríamos los valores de xb e yb. Sustituiríamos estos valores con las ecuaciones sxx, syy y sxy, los cuales podrían ser ya calculados en una segunda pasada.

El realizar estas dos pasadas es un proceso que lleva tiempo de ordenador, por lo que el programa utiliza un método por el cual se calculan los 5 conceptos en una sola pasada. La nueva forma de las ecuaciones se puede observar entre las líneas 3060 - 3100.

El programa utiliza estos 5 conceptos para calcular la ecuación de la recta de regresión (líneas 3150 - 3450). Un ejemplo de la salida de resultados se muestra en la Figura FR2. Los resultados se generan utilizando los mismos datos de entrada utilizados en "REPRESENTACION DE DATOS", estando determinada la estatura en metros (con 2 decimales) y la edad en años, para los 30 escolares analizados.

	Term.Ind.	Pendiente
Coef	0.72059301	.050333957
Err	.049844433	.0044082279
T	14.45684	11.418184

	Recta	Residuo
DF	1	28
SS	1.6234715	0.3486652
MSS	1.6234715	.012452329
F	130.37493	

FIGURA FR2. **Un ejemplo de la salida del programa REGRESION**

La Figura FR2 muestra que la ecuación de la recta de regresión que enlaza la estatura con la edad es:

ESTATURA = 0,72 + 0,05 * EDAD

Los errores en el término independiente y en la pendiente con 0.05 y 0.0044 respectivamente y el estadístico T (STUDENT) son indicativos de que ambos valores son significativamente distintos de cero. La mitad inferior de la Figura FR2 es una tabla de Análisis de Variancia. El significado de las abreviaturas es el siguiente:

DF - GRADOS DE LIBERTAD
SS - SUMA DE CUADRADOS
MSS - MEDIA DE LA SUMA DE CUADRADOS
F - STADISTICO F DE SNEDECOR

El valor de F obtenido es indicativo de que existe una fuerte relación entre la estatura y la edad.

PROGRAMA PR1. **REGRESION**

```
  1 GO SUB 7000
 10 LET t=24
500 PRINT ,,TAB 11;"Regresion",,,"Los datos van a ser
    introducidospor: Teclado o Cassette? (T/C)"
510 INPUT z$
520 IF CODE z$>96 THEN  LET z$=CHR$ (CODE z$-32)
530 IF z$<>"T" AND z$<>"C" THEN  BEEP 1,12: GO TO 510
540 GO TO 550+200*(z$="T")
550 PRINT ,,"Introduce el nombre del archivo del cassette"
560 INPUT z$: IF LEN z$>10 THEN  BEEP 1,12: GO TO 560
```

```
 570 PRINT AT 21,0; FLASH 1;"REALIZANDO CARGA DE LOS DATOS"
 580 LOAD z$ DATA x()
 590 PRINT AT 19,0;z$;"      CARGA CORRECTA           "
 600 LET n=x(1,1)
 610 GO TO 1100
 750 PRINT ,,"Introduce el numero de pares de datos"
 760 INPUT n: IF n<1 THEN  BEEP 1,12: GO TO 760
 770 DIM x(2,n+1)
 780 PRINT "Introduce los valores de x e y  de cada par de datos
     por orden",,,"INDICE";TAB 14;"X";TAB 26;"Y"
 790 FOR i=2 TO n+1
 795 PRINT i-1;TAB 14;
 800 INPUT x(1,i): PRINT x(1,i);TAB 26;
 810 INPUT x(2,i): PRINT x(2,i),
 820 NEXT i
 830 LET x(1,1)=n
1000 CLS
1010 PRINT TAB 11;"REGRESION"
1020 PRINT ,,,,TAB 14;"Menu"
1030 PRINT ,,"Revisar datos";TAB t;"R","Listar datos";TAB t;"L",
     "Corregir un valor";TAB t;"C","Calcular la regresion";
     TAB t;"F","Grabar datos";TAB t;"S","Verificar los datos";
     TAB t;"V","Detener el programa";TAB t;"H"
1040 INPUT z$: IF CODE z$>96 THEN  LET z$=CHR$ (CODE z$-32)
1050 GO TO 1100+400*(z$="R")+900*(z$="L")+1400*(z$="C")+1900*
     (z$="F")+3900*(z$="S")+4400*(z$="V")+4900*(z$="H")
1100 PRINT ,,"Pulsa tecla para volver al menu"
1110 PAUSE 0
1120 GO TO 1000
1500 PRINT ,,TAB 10;"REVISION DE DATOS",,,"Indice";TAB 14;"X";
     TAB 26;"Y"
1510 FOR i=2 TO n+1
1520 PRINT i-1;TAB 14;x(1,i);TAB 26;x(2,i)
1530 NEXT i
1540 GO TO 1100
2000 PRINT ,,TAB 11;"LISTADO DE DATOS",,,
2010 LPRINT ,,"Indice";TAB 14;"X";TAB 26;"Y"
2020 FOR i=2 TO n+1
2030 LPRINT i-1;TAB 14;x(1,i);TAB 26;x(2,i)
2040 NEXT i
2050 GO TO 1100
2500 PRINT ,,TAB 8;"CORRECCION DE DATOS"
2510 PRINT ,,"Introduce el indice del valor a cambiar"
2520 INPUT a: IF a<1 OR a>n THEN  BEEP 1,12: GO TO 2520
2530 PRINT ,,"Los  valores son:";TAB 8;x(1,a+1);TAB 20;x(2,a+1)
2540 PRINT ,,"Introduce los nuevos valores de X e Y por orden"
2550 INPUT x(1,a+1),x(2,a+1)
2560 GO TO 1100
3000 PRINT ,,TAB 5;"CALCULO DE LA REGRESION"
3010 PRINT ,,"Quieres una copia en la impreso ra? (S/N)"
3020 INPUT y$: LET y$=CHR$ (CODE y$-32*(y$)"£")): IF y$<>"S"
     AND y$<>"N" THEN  BEEP .5,24: GO TO 3020
3030 IF y$="S" THEN  CLS
3040 LET xb=0: LET yb=0: LET sxx=0: LET syy=0: LET sxy=0
3050 FOR i=1 TO n
3060 LET sxx=sxx+(i-1)*(x(1,i+1)-xb)*(x(1,i+1)-xb)/i
3070 LET syy=syy+(i-1)*(x(2,i+1)-yb)*(x(2,i+1)-yb)/i
3080 LET sxy=sxy+(i-1)*(x(1,i+1)-xb)*(x(2,i+1)-yb)/i
3090 LET xb=((i-1)*xb+x(1,i+1))/i
3100 LET yb=((i-1)*yb+x(2,i+1))/i
```

```
3110 NEXT i
3150 LET b=sxy/sxx
3160 LET seb=((syy*sxx-sxy*sxy)/(n-2))^.5/sxx
3170 LET a=yb-b*xb
3180 LET sea=(syy/(n*(n-1))+xb*seb*seb)^.5
3300 LET s1=5: LET s2=19
3310 PRINT ,,TAB s1;"Term.Ind.";TAB s2;"Pendiente"
3320 PRINT ,,,,"Coef ";a;TAB s2;b
3330 PRINT ,,"Err  ";sea;TAB s2;seb
3340 PRINT ,,"T";TAB s1;a/sea;TAB s2;b/seb
3400 PRINT ,,,,,,TAB s1;"Recta";TAB s2;"Residuo"
3410 PRINT ,,"DF";TAB s1;"1";TAB s2;n-2
3420 LET mss=b*b*sxx: LET rss=syy-mss
3430 PRINT ,,"SS";TAB s1;mss;TAB s2;rss
3440 PRINT ,,"MSS";TAB s1;mss;TAB s2;rss/(n-2)
3450 PRINT ,,"F";TAB s1;mss*(n-2)/rss
3460 IF y$="S" THEN  COPY : CLS
3500 PRINT ,,"Quieres grabar los residuos y   los valores
     calculados? (S/N)"
3510 INPUT y$: LET y$=CHR$ (CODE y$-32*(y$)"`")): IF y$<>"S"
     AND y$<>"N" THEN  BEEP .5,24: GO TO 3020
3520 IF y$="N" THEN  GO TO 1100
3530 DIM y(2,n+1)
3540 FOR i=1 TO n+1
3550 LET y(1,i)=x(1,i): LET y(2,i)=x(2,i)
3560 NEXT i
3570 FOR i=1 TO n
3580 LET x(2,i+1)=a+b*y(1,i+1)
3590 LET x(1,i+1)=y(2,i+1)-x(2,i+1)
3595 NEXT i
3600 PRINT ,,"Introduce el nombre de DATOS"
3610 INPUT z$: IF LEN z$>10 THEN  BEEP 1,12: GO TO 3610
3620 PRINT AT 21,0; FLASH 1;"  REALIZANDO GRABACION        "
3630 SAVE z$ DATA x()
3640 PRINT AT 21,0;z$;"    GRABACION CORRECTA  "
3650 PRINT ,,"Por favor rebobina la cinta     para realizar
     la verificacion",,,"Pulsa una tecla cuando termine"
3660 PAUSE 0
3670 PRINT AT 21,0; FLASH 1;"  REALIZANDO LA VERIFICACION  "
3680 VERIFY z$ DATA x()
3690 PRINT AT 19,0;z$;"   VERIFICACION CORRECTA  "
3700 FOR i=1 TO n+1
3710 LET x(1,i)=y(1,i): LET x(2,i)=y(2,i)
3720 NEXT i
3800 GO TO 1100
5000 PRINT ,,TAB 11;"GRABAR DATOS",,,"Introduce el nombre"
5010 INPUT z$: IF LEN z$>10 THEN  BEEP 1,12: GO TO 5010
5020 PRINT AT 21,0; FLASH 1;"REALIZANDO LA GRABACION"
5030 SAVE z$ DATA x()
5040 PRINT AT 21,0;z$;"   GRABACION CORRECTA  "
5050 GO TO 1100
5500 PRINT ,,TAB 10;"VERIFICAR DATOS",,,"Introduce el nombre"
5510 INPUT z$: IF LEN z$>10 THEN  BEEP 1,12: GO TO 5510
5520 PRINT AT 21,0; FLASH 1;"REALIZANDO LA VERIFICACION"
5530 VERIFY z$ DATA x()
5540 PRINT AT 19,0;z$;" VERIFICACION CORRECTA"
5550 GO TO 1100
```

```
6000 PRINT ,,"PROGRAMA DETENIDO"
6010 STOP
7000 PRINT ,,"Este programa realiza calculos  de regresiones,
     siempre y cuandose introduzcan mas de tres pares de datos
     y todos los valores diferentes"
7010 PAUSE 0
7020 RETURN
```

Fe de erratas:
La línea 3510 del programa debe ser:

```
3510 INPUT y$: LET y$=CHR$ (CODE y$-32*(y$)"£")): IF y$()"S"
     AND y$()"N" THEN  BEEP .5,24: GO TO 3020
```

9. HISTOGRAMA

Este es el tercero de los programas relacionados con el análisis estadístico de datos, siendo los otros dos "REPRESENTACION DE DATOS" y "REGRESION".

Los 3 programas han sido realizados con el fin de su utilización conjunta, de tal forma que cada uno de ellos conforma un aspecto diferente del análisis de un determinado volumen de datos. Una particularidad es que los datos introducidos mediante teclado en el programa pueden ser GRABADOS en cinta y posteriormente CARGADOS en otro de los Programas, con lo cual se evita una innecesaria duplicación de esfuerzos.

Una diferencia importante entre HISTOGRAMA y los otros dos programas mencionados es que sólo analiza una variable, no dos. Asimismo, cuando los datos son transferidos a HISTOGRAMA desde otro de los programas, el computador pregunta al usuario cuál de las variables desea seleccionar para su análisis. La variable que no es seleccionada es dejada a un lado y puede ser llamada en una etapa posterior mediante la opción apropiada del MENU principal. Cuando los datos se introducen en el programa mediante el teclado sólo se pueden introducir datos de una variable y por lo tanto no pueden servir como datos de entrada a los otros 2 programas del "paquete" para análisis de datos.

Las opciones disponibles en el programa HISTOGRAMA pueden verse en la Figura FI1. Sólo la opción titulada "DIBUJO DEL HISTOGRAMA" es interesante explicarla, siendo el resto de ellas suficientemente autoexplicativas.

```
                    HISTOGRAMA

                       Menu

Revisar los datos                R
Listar los datos                 L
Corregir los datos               C
Dibujar el histograma            D
Parar el programa                H
```

FIGURA FI1. **MENU para HISTOGRAMA**

Cuando se elige la opción "DIBUJO DEL HISTOGRAMA" el programa dibuja los valores más pequeño y mayor de la variable y pregunta al usuario qué LIMITES INFERIOR Y SUPERIOR elige para el Histograma, así como el número de INTERVALOS (que luego darán lugar a rectángulos de mayor o menor longitud) elegido. Así el usuario tiene la oportunidad de estudiar todos o parte de los datos y puede elegir números "redondos" para los límites de cada intervalo del histograma. El programa redondea el número de intervalos al número más cercano a 4 debido a que escribe en el eje vertical del histograma sólo 5 valores (es decir, delimita 4 intervalos).

Limite inferior	Veces
-0.15	1
-0.1	4
-0.05	4
0	5
.05	1
0.1	10
0.15	2
0.2	1

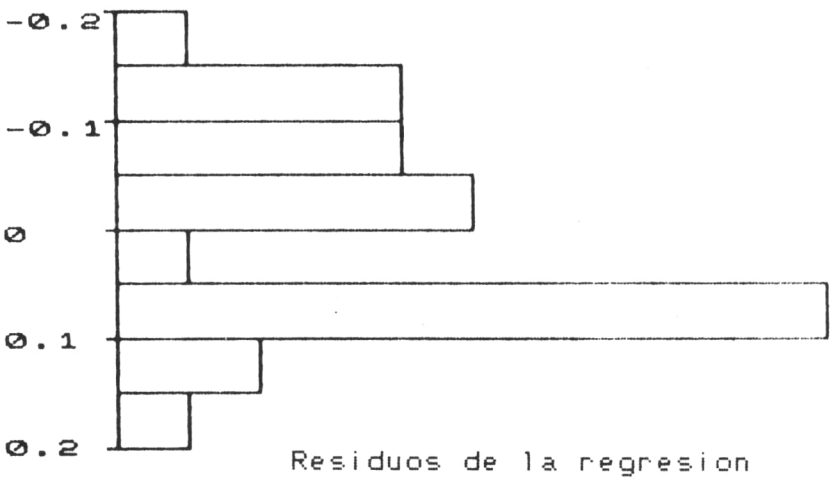


FIGURA FI2. **Un ejemplo de los resultados de HISTOGRAMA**

PROGRAMA PI1. **HISTOGRAMA**

```
 10 LET t=26
500 PRINT ,,TAB 11;"HISTOGRAMA",,,"Introduces los datos por
    teclado o los cargas del cassete        (T o C)?"
```

```
 510 INPUT y$
 520 IF CODE y$>96 THEN  LET y$=CHR$ (CODE y$-32)
 530 IF y$<>"T" AND y$<>"C" THEN  BEEP 1,12: GO TO 510
 540 GO TO 550+200*(y$="T")
 550 PRINT ,,"Introduce el nombre del fichero a cargar"
 560 INPUT z$: IF LEN z$>10 THEN  BEEP 1,12: GO TO 560
 570 PRINT AT 21,0; FLASH 1;"Cargando"
 580 LOAD z$ DATA x()
 590 PRINT AT 19,0;z$;"Programa cargado"
 600 LET n=x(1,1)
 610 PRINT ,,"Que variable (X o Y)?"
 620 INPUT z$: LET z$=CHR$ (CODE z$-32*(z$>"£")):
     IF z$<>"X" AND z$<>"Y" THEN  BEEP 1,12: GO TO 620
 630 LET w=1+(z$="Y")
 640 GO TO 1000
 750 PRINT ,,"Numero de datos a introducir"
 760 INPUT n: IF n<1 THEN  BEEP 1,12: GO TO 760
 770 DIM x(2,n+1)
 780 PRINT "Introduce los valores en orden",,,"Indice","Valor"
 790 FOR i=2 TO n+1
 800 PRINT i-1,: INPUT x(1,i): PRINT x(1,i)
 810 NEXT i
 820 LET x(1,1)=n
 830 LET w=1
1000 CLS
1010 PRINT TAB 11;"HISTOGRAMA"
1020 PRINT ,,,TAB 14;"Menu"
1030 PRINT ,,"Revisar los datos";TAB t;"R","Listar los datos";
     TAB t;"L","Corregir los datos";
     TAB t;"C","Dibujar el histograma";
     TAB t;"D","Parar el programa";TAB t;"H"
1035 IF y$="C" THEN  PRINT "Seleccionar otra variable";TAB t;"S"
1040 INPUT z$: IF CODE z$>96 THEN  LET z$=CHR$ (CODE z$-32)
1050 GO TO 1060+440*(z$="R")+940*(z$="L")+1440*(z$="C")+1940*
     (z$="D")+4940*(z$="H")
1060 IF z$="S" AND y$="C" THEN  LET w=2-(w=2):
     PRINT ,,"Otra variab. ha sido selec."
1100 PRINT ,,"Pulsa tecla para volver al menu"
1110 PAUSE 0
1120 GO TO 1000
1500 PRINT ,,TAB 10;"Revision de datos",,,"Indice","Valor"
1510 FOR i=2 TO n+1
1520 PRINT i-1,x(w,i)
1530 NEXT i
1540 GO TO 1100
2000 PRINT ,,TAB 11;"Imprime los datos",,,
2010 LPRINT ,,"Indice","Valor"
2020 FOR i=2 TO n+1
2030 LPRINT i-1,x(w,i)
2040 NEXT i
2050 GO TO 1100
2500 PRINT ,,TAB 10;"Datos correctos"
2510 PRINT ,,"Introduce el indice del valor a alterar"
2520 INPUT a: IF a<1 OR a>n THEN  BEEP 1,12: GO TO 2520
2530 PRINT ,,"Valor actual = ";x(w,a+1)
2540 PRINT ,,"Introduce el nuevo valor"
2550 INPUT x(w,a+1)
2560 GO TO 1100
3000 PRINT ,,TAB 11;"HISTOGRAMA"
3010 LET xn=x(w,2)
3020 LET xx=x(w,2)
3030 FOR i=2 TO n+1
```

```
3040 IF xn>x(w,i) THEN  LET xn=x(w,i)
3050 IF xx<x(w,i) THEN  LET xx=x(w,i)
3060 NEXT i
3100 PRINT ,,"Valor minimo = ";xn
3110 PRINT ,,"Valor maximo = ";xx
3120 PRINT ,,"Introduce el min. y el max.
     valor del histograma"
3130 INPUT xn,xx: IF xn>=xx THEN  BEEP .5,24: GO TO 3120
3140 PRINT ,,"Introduce numero de rectangulos requeridos
     (un multiplo de 4)"
3150 INPUT nb: IF nb<1 THEN  LET nb=1
3160 LET nb=4*INT ((nb+3)/(4))
3170 LET xs=(xx-xn)/nb
3200 PRINT ,,"Da nombre al histograma"
3210 INPUT t$
3220 IF t$="" THEN  GO TO 3280
3230 PRINT ,,"Linea y columna donde debe      aparecer el titulo"
3240 INPUT tl,tc
3250 IF tl<0 OR tl>21 OR tc<0 OR tc>31 THEN
     BEEP .5,24: GO TO 3240
3280 PRINT "Quieres imprimirlo(s/n)"
3290 INPUT c$: IF c$<>"s" AND c$<>"n" THEN  GO TO 3280
3300 PRINT ,,"Limite inferior","Veces",,,: IF c$="s" THEN
     LPRINT "Limite inferior","Veces",,,
3310 DIM b(nb)
3320 LET cl=0: LET ch=0
3330 FOR j=2 TO n+1
3340 LET i=INT ((x(w,j)-xn)/xs)+1
3350 IF i<1 THEN  LET cl=cl+1: GO TO 3380
3360 IF i>nb THEN  LET ch=ch+1: GO TO 3380
3370 LET b(i)=b(i)+1
3380 NEXT j
3390 FOR i=1 TO nb
3400 PRINT xn+i*xs,b(i): IF c$="s" THEN  LPRINT xn+i*xs,b(i)
3410 NEXT i
3420 IF c$="s" THEN  LPRINT
3430 IF ch*cl=0 THEN  GO TO 3460
3440 PRINT ,,cl;"menor que limite inferior",ch;
     "mayor que limite superior"
3450 IF c$="s" THEN  LPRINT cl;"menor que limite inferior",ch;
     "mayor que limite superior"
3460 PRINT ,,"Pulsa una tecla para continuar"
3470 PAUSE 0
3500 CLS : IF c$="s" THEN  LPRINT : LPRINT
3510 LET x=35: LET y=11
3520 PLOT x,y
3530 DRAW -3,0
3540 DRAW 3,0
3550 FOR i=1 TO 4
3560 DRAW 0,41
3570 DRAW -3,0
3580 DRAW 3,0
3590 NEXT i
3700 LET a=0
3710 FOR i=xn TO 1.01*xx STEP (xx-xn)/4
3720 LET z$=STR$ i+"     "
3730 PRINT AT a,0;z$( TO 4)
3740 LET a=a+5
3750 NEXT i
4000 LET mnb=0
4010 FOR i=1 TO nb
4020 IF mnb<b(i) THEN  LET mnb=b(i)
```

```
4030 NEXT i
4050 PLOT x,175
4060 FOR i=1 TO nb
4070 LET l=(255-x)*b(i)/mnb
4080 LET d=INT (164*i/nb)-INT (164*(i-1)/nb)
4090 DRAW l,0
4100 DRAW 0,-d
4110 DRAW -l,0
4120 NEXT i
4200 IF t$="" THEN  GO TO 4220
4210 PRINT AT tl,tc;t$
4220 IF c$="s" THEN  COPY
4300 PAUSE 0
4310 GO TO 1000
6000 PRINT ,,"Programa detenido"
6010 STOP
```

10. DIBUJO DE UNA FUNCION

Este programa está diseñado para dibujar la GRAFICA de una ECUACION definida por el propio usuario. El usuario puede seleccionar los límites entre los cuales desea dibujar la ecuación y además copiar dicho dibujo mediante la impresora. Un ejemplo de la salida del programa se muestra en la Figura FT1.

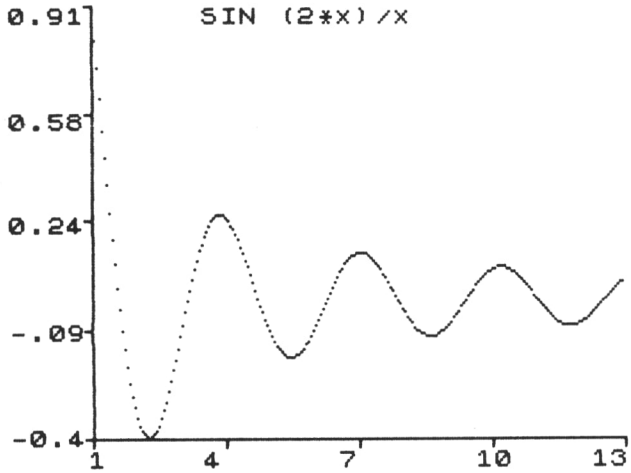


FIGURA FT1. GRAFICA de SIN (2 * x)/x para 1 ≦ x ≦ 13

Una vez ejecutado (RUN), el programa pregunta al usuario la ecuación (función de x) que desea dibujar. El programa no comprueba la sintáxis de la ecuación, debido a lo cual se ha de tener cuidado de no introducir caracteres ilegales, que los paréntesis sean pares, etcétera. El usuario debe teclear después de los límites mínimo y máximo del valor de x (abcisas) que se desean.

El programa también pregunta al usuario por el valor de las ASINTOTAS si es que existen. Una ASINTOTA es un valor de x para el cual la ecuación adopta un valor ± ∞ (más, menos INFINITO). La ecuación y = LOG x, por ejemplo, tiene una asíntota en x = Ø porque LOG Ø = - ∞.D. De igual forma la ecuación y = TAN x tiene como asíntotas x = π/2, 3*π/2, s * π/2, etc.

Si la ecuación elegida por el usuario tiene UNA o MAS asíntotas el programa pregunta por la tolerancia en el cálculo, para así evitar el cálculo de la ecuación para valores de x que están cercanos a las asíntotas (como pasaría en el caso en que la tolerancia fuera CERO). En otras palabras, si hay una asíntota en x = 1 y la tolerancia es 0,01, el programa no calculará el valor de la ecuación para valores de x comprendidos entre 0,99 y 1,01.

El programa calcula la ecuación para 221 valores de x dentro del intervalo definido, eligiendo unos límites adecuados para el dibujo de los ejes coordenados (x e y). Las subrutinas que determinan los valores dentro de cada intervalo (líneas 2200 a 2450) utilizan un bucle 5 veces consecutivas, una para cada intervalo. El valor máximo calculado por el bucle es un poco mayor que el máximo requerido debido a la necesidad de que los errores acumulados mediante el redondeo no hagan terminar el dibujo del cuarto intervalo dentro del espacio dentro del espacio del quinto intervalo. La siguiente subrutina ilustra lo que ocurriría en caso contrario:

```
10  INPUT J
20  FOR i = 0 TO J STOP J/4
30  PRINT i
40  NEXT i
```

Si el valor de j = 1,01 los resultados del programa serán:

0,0.2525, 0,505, 0.7575, 1.01

Si el valor de j = 1.02. Sin embargo, el programa escribirá:

0,0.255, 0.51, 0.765

En esta segunda ocasión sólo se escriben 4 resultados porque los errores de redondeo se han acumulado al valor de STEP, añadiéndose al valor de la variable i en cada STEP.

Después de dibujar la GRAFICA, el programa indica al usuario que ha de introducir las coordenadas para el comando PRINT AT, utilizado para dibujar la función (línea 2620). Si el usuario no está satisfecho de la posición elegida puede cambiarla simplemente escribiendo sobre ella mediante la utilización sucesiva de OVER 1 (línea 2650).

PROGRAMA PT1. DIBUJO DE UNA FUNCION

```
100 PRINT TAB 6;"DIBUJO DE UNA FUNCION",,,
    "Introduce f(x) para su represen-tacion."
110 INPUT f$: PRINT ,,"La funcion es ";f$
120 PRINT ,,"Introduce los valores minimo y  maximo de la
    funcion para poder representarla",,,,
130 PRINT TAB 0;"X minima = ",: INPUT xn: PRINT xn
140 PRINT TAB 0;"X maxima = ",: INPUT xx: PRINT xx
150 PRINT ,,"Introduce el no. de asintotas"
160 INPUT as: IF as<0 THEN  BEEP 1,12: GO TO 160
170 IF as=0 THEN  GO TO 300
180 DIM a(as)
190 PRINT ,,"Introduce los valores X en cada asintota por orden"
200 FOR i=1 TO as
210 INPUT a(i)
220 NEXT i
230 PRINT ,,"Introduce la tolerancia periti- da en las
    asintotas (ej .01)"
240 INPUT tol: IF tol<0 THEN  BEEP 1,12: GO TO 240
300 PRINT AT 21,0; FLASH 1;"Realizando los calculos."
310 DIM x(221): DIM y(221)
320 LET i=1
330 FOR x=xn TO xx STEP (xx-xn)/220
340 IF as=0 THEN  GO TO 400
350 FOR j=i TO as
360 IF ABS (x-a(j))<tol THEN  GO TO 430
370 NEXT j
400 LET x(i)=x
410 LET y(i)=VAL f$
420 LET i=i+1
430 NEXT x
440 LET n=i-1
500 LET yn=y(1)
510 LET yx=y(1)
520 FOR i=1 TO n
530 IF yn>y(i) THEN  LET yn=y(i)
540 IF yx<y(i) THEN  LET yx=y(i)
550 NEXT i
560 LET yn=yn*.99
570 LET yx=yx*1.01
580 PRINT AT 21,0,,
1000 PRINT ,,"Quieres una copia en la impreso-ra del dibujo de la
     funcion?    (S/N)"
1010 INPUT c$: LET c$=CHR$ (CODE c$-32*(CODE c$>96))
1020 IF c$<>"S" AND c$<>"N" THEN  BEEP 1,12: GO TO 1010
2000 CLS
2010 LET x=35: LET y=11
2020 DRAW  INVERSE 1; OVER 1;x,y
2030 FOR i=1 TO 4
2040 DRAW 0,41
2050 DRAW -3,0
2060 DRAW 3,0
2070 NEXT i
```

```
2080 DRAW 0,-164
2090 DRAW -3,0
2100 DRAW 3,0
2110 DRAW 0,-3
2120 DRAW 0,3
2130 FOR i=1 TO 4
2140 DRAW 55,0
2150 DRAW 0,-3
2160 DRAW 0,3
2170 NEXT i
2180 DRAW -228,0
2200 LET a=4
2210 FOR i=xn TO xx STEP (xx-xn)/4
2220 LET z$=STR$ i
2230 IF LEN z$>4 THEN  LET z$=z$( TO 4)
2240 IF a=31 THEN  LET a=32-LEN z$
2250 PRINT AT 21,a;z$
2260 LET a=a+6+(a>9)
2270 NEXT i
2400 LET a=20
2410 FOR i=yn TO 1.01*yx STEP (yx-yn)/4
2420 LET z$=STR$ i+"  "
2430 PRINT AT a,0;z$( TO 4)
2440 LET a=a-5
2450 NEXT i
2500 FOR i=1 TO n
2510 LET x=35+220*(x(i)-xn)/(xx-xn)
2520 LET y=12+163*(y(i)-yn)/(yx-yn)
2530 PLOT x,y
2540 NEXT i
2600 INPUT "Coordenadas del titulo ";x;" ";y
2610 IF x<0 OR x>21 OR y<0 OR y>31 THEN  BEEP 1,12: GO TO 2610
2620 PRINT  OVER 1;AT x,y;f$
2630 INPUT "Es correcto? (S/N)";z$
2640 LET z$=CHR$ (CODE z$-32*(z$)"£")): IF z$<>"S" AND z$<>"N"
     THEN  BEEP 1,12: GO TO 2630
2650 IF z$="N" THEN  PRINT  OVER 1;AT x,y;f$: GO TO 2600
2660 IF c$="S" THEN  COPY
```

11. RENUMERADOR DE LINEAS

Un elemento diferencial del lenguaje de programación BASIC es el uso de números de línea para controlar el orden en el cual las líneas del programa son almacenadas y ejecutadas. Este elemento tiene la ventaja de realizar la edición de las líneas del programa de forma suficientemente clara para el usuario porque cada línea tiene su propio número como única referencia.

Sin embargo esto tiene también algunas desventajas de distinta importancia. Es una pérdida de espacio, puesto que la mayoría de los números de las líneas son utilizadas sólo durante el desarrollo del programa, no durante la ejecución. Segundo: el programador debe frecuentemente perder tiempo cambiando números de líneas en orden a crear suficiente espacio como para intercalar una nueva subrutina. Una tercera desventaja es el hecho de que los números de las líneas son utilizadas también para identificar los destinos de las instrucciones GOTO y GOSUB. Así se ha de tener gran cuidado al cambiar los números de las líneas y asegurarse de que todos los GOTO y GOSUB que llaman a dichas líneas se han cambiado también.

He escrito este RENUMERADOR DE LINEAS como una subrutina para hacer la tarea del cambio de los números de las líneas algo más fácil, aunque el usuario debe renumerar los GOTO y GOSUB. El principio por el que opera el programa es muy simple y se resume en el programa PL1, muy corto, el cual ha sido concebido para renumerarse el mismo comenzando en 8 mediante pasos de 4 en 4. Este programa trabaja mediante la restauración del primer byte del área de programa (y asimismo la dirección del primero de los dos bytes que se utilizan para almacenar el número de la primera línea) desde la variable del sistema PROG, almacenada en las direcciones 23635 y 23636 (ver páginas 173 a 176 del manual de Programación Basic para el ZX SPECTRUM). El nuevo número de línea es puesto en su lugar mediante las líneas 20 y 25. La línea 35 utiliza la longitud de la línea del programa que está almacenada en las 2 localizaciones siguientes a la línea numerada para calcular la dirección de la nueva línea del programa. Para prevenir que la nueva dirección no vaya más allá del final del área de variables (y asimismo no sea mayor que el valor de la variable del Sistema VARS, almacenada en las direcciones 23627 y 23628), el programa vuelve a la línea 20 para empezar la siguiente línea del programa.

```
 5 LET num=8
10 LET ss=4
15 LET add=PEEK 23635+256*PEEK 23636
20 POKE add,num/256
25 POKE add+1,num-256*INT (num/256)
30 LET num=num+ss
35 LET add=add+4+PEEK (add+2)+256*PEEK (add+3)
40 IF add < PEEK 23627+256*PEEK 23628 THEN GO TO 20
```

PROGRAMA PL1. Un programa que se renumera solo

Este programa es sólo un artificio, porque sólo trabaja si la línea número 20 conserva el mismo número durante su ejecución, pues de otra manera el GOTO de la línea 40 se dirige a una línea errónea. Esto sirve para ilustrar cómo un lenguaje de programación podría tratar mejor los números de las líneas y los números de destino de las sentencias GOTO y GOSUB como cosas diferentes (los mejores lenguajes evitan la utilización de comandos tipo GOTO obviando de esta manera el problema).

La SUBRUTINA completa (PL2) tiene como objetivo RENUMERAR una parte de un programa. Ocupa desde la línea 9000, por lo que, si se evita en los programas utilizar un número de línea mayor de 8999, puede fácilmente ser MEZCLADO (MERGE) con cualquier programa existente.

Llamamos al programa mediante RUN 9000 y él entonces solicita los números de la primera y la última línea que han de ser renumerados sólo aceptará respuestas lógicas; por ejemplo; líneas con números negativos o con números de línea mayores de 8999 serán rechazados. El programa explora la memoria para encontrar los números de las líneas que han de ser renumeradas y calcula el incremento máximo permisible (o tamaño del salto) entre dos nuevos números de línea. En este punto el programa indica al usuario que debe seleccionar un nuevo número de línea de comienzo y el tamaño del intervalo y, cuidando de que ambos caigan entre los límites permitidos procede a la renumeración.

Debemos poner énfasis al indicar que esta SUBRUTINA **NO** renumera los comandos GOTO y GOSUB de destino. El llevar esto a cabo podría ser casi imposible de escribir mediante el lenguaje BASIC. La secuencia de operaciones de un programa adecuado un CODIGO MAQUINA podría ser:

1. Almacenar las direcciones y números de destino de todos los GOSUB y GOTO del programa en un área de trabajo adecuada.
2. Renumerar cada línea por orden, comprobando el número de línea antiguo en el área de trabajo.
3. Cuando se encuentra un número igual, convertir el nuevo número de línea a caracteres y volver a escribir los caracteres del número de destino.
4. Si los números nuevo y antiguo difieren en longitud se debe ampliar o reducir en consecuencia el área de programa. (Por ejemplo, si el número antiguo era 95 y el nuevo es 140, el área de programa podría ampliarse en un byte de longitud debido a que 95 tiene 2 caracteres y 140 tiene 3). Una vez hecho esto, las direcciones están almacenadas en la memoria de forma correcta.
5. Convertir el nuevo número de línea a la forma de punto flotante y reescribir en su forma numérica el número de destino, el cual está escondido en los 5 bytes (más un byte separador) acompañando al carácter del número.
6. Borrar todo lo introducido en el espacio de trabajo, de tal forma que los números no se renumeren accidentalmente más de una vez.

PROGRAMA PL2. RENUMERADOR DE LINEAS

```
9000 GO SUB 9970
9010 PRINT "Introduce la primera y ultima
     linea del bloque a renumerar"
9020 INPUT a,b
9030 IF a<0 OR b<=a OR b>8999 OR a<>INT a OR b<>INT b THEN
     BEEP .2,24: GO TO 9020
9040 GO SUB 9800
9050 GO SUB 9970
9060 PRINT "numero de comienzo actual= ";num
9070 LET snum=onum: LET sadd=add
9080 LET a=b: LET m=n
9090 GO SUB 9800
9100 PRINT "Numero de final actual   = ";onum
9110 PRINT ,,"El nuevo comienzo debe ser>";TAB 27;snum
9120 PRINT "El nuevo final debe ser< ";TAB 27;num
9130 PRINT ,,"No lineas a renumerar = ";TAB 27;n-m
9140 LET ss=INT ((onum-snum)/(n-m))
9150 PRINT "El incremento debe ser <= ";TAB 27;ss
9200 PRINT ,,"Introduce la nueva linea de
     comienzo y el incremento "
9210 INPUT a,b
9220 IF a<=snum OR b<1 OR b>ss OR a<>INT a OR b<>INT b THEN
     BEEP .2,24: GO TO 9210
9300 PRINT ,,"nuevo comienzo = ";TAB 27;a
9310 PRINT "Incremento     = ";TAB 27;b
9320 FOR i=1 TO n-m
9330 POKE sadd,INT (a/256)
9340 POKE sadd+1,a-256*INT (a/256)
9350 LET sadd=sadd+4+PEEK (sadd+2)+256*PEEK (sadd+3)
9360 LET a=a+b
9370 NEXT i
9380 PRINT TAB 4; FLASH 1;"Renumeracion terminada"
9390 STOP
9800 LET onum=0: LET oadd=0: LET n=0
9810 LET add=PEEK 23635+256*PEEK 23636
9820 LET num=256*PEEK add+PEEK (add+1)
9830 IF num=a AND a<>b THEN  RETURN
9840 IF num>a THEN  RETURN
9850 LET onum=num: LET oadd=add
9860 LET add=add+4+PEEK (add+2)+256*PEEK (add+3)
9870 LET n=n+1
9880 GO TO 9820
9970 CLS
9980 PRINT TAB 9;" RENUMERADOR ",,,
9990 RETURN
```

12. CARGA DE RUTINAS EN CODIGO MAQUINA (HEX LOADER)

Este programa es muy utilizado para almacenar subrutinas cortas en CODIGO MAQUINA tal como las que se publican periódicamente en las revistas sobre computadores, debido a lo cual yo he incluido una pareja de subrutinas encadenadas como muestra. La primera rutina renumera un programa (excepto los números de línea de destino en GOSUB y GOTO) y la segunda graba los resultados y las vuelve a llamar. Este programa no es demasiado cómodo para desarrollar subrutinas en código máquina, puesto que no tiene características sofisticadas. Un programa más cómodo para realizar este trabajo es el EDITOR EN CODIGO MAQUINA, que también se incluye en este volumen.

El MENU de opciones que controla el programa se encuentra en la línea 500 y se ilustra en la Figura FH1. El usuario selecciona una opción del MENU, debido a lo cual el programa salta a la subrutina apropiada desde la línea 520.

MENU

DE TECLADO A MEMORIA	K
SENTENCIAS REM A MEMORIA	R
SALIDA EN HEXADECIMAL	D
HEXADECIMAL A REM	H
PARADA	S

Figura FH1. **El MENU de opciones de HEX LOADER**

MANEJO DE LAS DIRECCIONES DE REENTRADA ("RETURN")

La opción "DE TECLADO A MEMORIA" permite al usuario introducir código de máquina en HEXADECIMAL mediante teclado. El programa pregunta en primer lugar, utilizando la subrutina de la línea 6000, por la dirección en la cual se almacena la forma "absoluta" del código. El usuario debe introducir una dirección que sea mayor que el RAMTOP.

Si la dirección está bajo el RAMTOP el programa pide la confirmación del dato. Si la respuesta es "s" (se contesta "sí") entonces el programa convierte el código hexadecimal introducido posteriormente por el usuario en su forma numérica y lo direcciona a la memoria comenzando en la dirección que se introdujo en un principio.

Es de resaltar lo que sucede cuando la respuesta es "N" (se contesta "NO"). El programa está en una subrutina, pero ahora no puede volver (RETURN) al programa principal porque el usuario no desea seguir. Si a continuación se hiciera GOTO 600, se podría dejar la dirección de retorno -RETURN- (por ejemplo: el número de la línea en la que debería haberse constituido el comando RETURN), en el almacenamiento de GOSUB. Esta situación se produce porque no hay sitio suficiente en el área de BASIC del ZX SPECTRUM para que una subrutina retorne (RETURN) a cualquiera de los 2 puntos de reentrada dependiendo de una decisión tomada dentro de una subrutina.

El programa soslaya este problema direccionando un nuevo numero de línea de retorno (RETURN), la 599, en el almacenamiento de GOSUB y ejecutando entonces el RETURN. El nuevo número de línea de retorno (nr) se toma como 599 porque el software del ROM incrementa el número. Así el programa continúa desde la línea 600.

La forma del almacenamiento de GOSUB puede inspeccionarse utilizando el bucle de las líneas 9000 - 9030. Este bucle no es parte del programa principal y se puede omitir si se desea. Los valores que se indican en el listado del programa se refieren a una máquina de 48K. Para utilizar una de 16K se debe cambiar 65367 por 32599.

Para ver el almacenamiento de GOSUB debemos primeramente tener absoluta certeza de que no existen direcciones de RETURN innecesarias dentro de dicho almacenamiento, para lo cual introducimos por teclado RETURN varias veces hasta que aparezca el mensaje:

7. RETURN WITHOUT GOSUB, Ø : 1

Ahora se debe introducir RUN 9000 y a continuación debe aparecer la siguiente TABLA:

65367	62
65366	?
65365	19
65364	3
65363	27
65362	118
65361	127
65360	88
.	.
.	.

El valor de la dirección 65366 varía con la tarea que la máquina ha estado realizando desde la ejecución del programa y aparece en la TABLA como "?". Esta es la manera en la que el almacenamiento aparece cuando no existen actualmente ningún número de líneas de retorno (RETURN).

Ahora introducimos una nueva línea:

9600 GOSUB 9000

y la llamamos mediante RUM 9600. Realizando esto situaremos la dirección de retorno (RETURN) 9600 en el almacenamiento de GOSUB. La tabla que aparecía en la línea 9000 tendrá ahora el siguiente aspecto:

65367	62
65366	?
65365	2
65364	37
65363	128
65362	19
65361	3
65360	27
.	.
.	.

Se han insertado 3 bytes en la tabla. El primero contiene un "2", lo que indica que a continuación hay 2 números de bytes, el segundo y tercer bytes contienen "37" y "128" respectivamente y componen conjuntamente el número de la línea de RETURN (ya que 37 * 256 + 128 = 9600).

Es dentro de aquellas localizaciones donde la subrutina direcciona la nueva línea de retorno (RETURN) requerida en los POKES de las líneas 9500 a 9540.

EL RESTO DE LAS OPCIONES

Las restantes opciones del programa son completamente autoexplicativas.

La opción "SENTENCIAS REM A MEMORIA" almacena el código hexadecimal existente en una sentencia REM (constituyendo la primera línea de un programa) en código absoluto y en una localización especificada por el usuario. El listado que se adjunta muestra la subrutina RENUMERADOR de LINEAS almacenado de esta manera.

La opción "SALIDA EN HEXADECIMAL" lista los códigos hexadecimales dentro de un intervalo de direcciones de memoria seleccionado por el usuario.

La opción "HEXADECIMAL A REM" convierte el código absoluto de un intervalo de direcciones a hexadecimal y las introduce dentro de sentencias REM. Esto puede utilizarse asimismo para copiar y modificar subrutinas en CODIGO MAQUINA en la ROM.

RENUMERADOR DE LINEAS

La tabla TH1 presenta una subrutina para renumerar un programa (excepto GOTO y GOSUB) en intervalos de 10 en 10.

01 0A 00		LD BC; 10
11 0A 00		LD DE; 10
2A 53 5c		LD HL, (23635)
72	NEXTLOC	LD (HL), D
23		INC HL
73		LD (HL), E
23		INC HL
D5		PUSH DE
5E		LD E, (HL)
23		INC HL
56		LD D, (HL)
23		INC HL
19		ADD HL, DE
EA 5B 4B 5C		LD DE, (23627)
E5		PUSH HL
A7		AND A
ED 52		SBC HL, DE
D1		POP DE
E1		POP HL
D0		RET NC
09		ADD HL, BC
EB		EX DE, HL
18 E7		JR, —25 NEXT LOC

TABLA TH1. Renumerador de líneas en código máquina (excepto sentencias GO TO y GO SUB)

GRABAR Y VOLVER A LLAMAR A LA PANTALLA

La tabla TH2 muestra 2 subrutinas para grabar en cinta la totalidad de la salida por pantalla, incluyendo el FICHERO de atributos en 25856, y para copiarla de nuevo en 16384. Se ha de tener cuidado de modificar el RAMTOP a 25855 o menos antes de que la rutina grabada en cassette sea ejecutada y de asegurarse para que las propias subrutinas no sean almacenadas entre 25856 y 32767.

Copia de la pantalla a 25856		Copia de 25856 a la pantalla	
21 00 40	LD HL; 16384	21 00 65	LD HL, 25856
11 00 65	LD DE, 25856	11 00 40	LD DE, 16384
01 00 1B	LD BC, 6912	01 00 1B	LD BC, 6912
ED B0	LD IR	ED B0	LD IR
C9	RET	C9	RET

TABLA TH2. **Grabado en cinta y almacenado de la pantalla incluyendo el archivo asignado a / desde 25856**

PROGRAMA PH1. HEX LOADER

```
  10 REM 010A00110A002A535C72237323D55E23562319ED5B4B5CE5A
         7ED52D1E1D009EB18E7
 500 CLS : PRINT ,,TAB 14;"MENU",,,,,"De teclado     a memoria";
     TAB 31;"KHexad. de REM a memoria";TAB 31;
     "RSalida en hexadecimal (s=STOP)";TAB 31;"DHexad. a REM";
     TAB 31;"HParar el programa;";TAB 31;"S"
 510 INPUT z$: LET z$=CHR$ (CODE z$-32*(z$>"£"))
 520 GO TO 600 +400*(z$="K")+900*(z$="R")+1400*(z$="D")+1900*
     (z$="H")+2400*(z$="S")
 599 REM
 600 INPUT "Pulsa ""ENTER"" para continuar";z$
 610 GO TO 500
1000 PRINT ,,TAB 2;"Hexad. de teclado a memoria"
1010 GO SUB 6000
1020 PRINT ,,"Introduce hexadec. ",,,
1030 INPUT z$: IF z$="" THEN  GO TO 600
1040 LET l=LEN z$
1050 IF l<>2*INT (l/2) THEN  GO TO 1440
1055 LET y$=z$
1060 FOR i=1 TO l
1066 LET y$(i)=CHR$ (CODE z$(i)-32*(z$(i)>"£"))
1070 LET z$(i)=CHR$ (CODE z$(i)-32*(z$(i)>"£")+7*(z$(i)<"A"))
1080 IF z$(i)<"7" OR z$(i)>"F" THEN  GO TO 1400
1090 NEXT i
1100 PRINT a,y$
1110 FOR i=1 TO l STEP 2
1120 POKE a,16*CODE z$(i)+CODE z$(i+1)-935
1130 LET a=a+1
```

```
1140 NEXT i
1150 GO TO 1030
1400 BEEP .2,24
1410 IF i=LEN z$ THEN  GO TO 1440
1420 PRINT ,,z$(1 TO i); FLASH 1;"?";z$(i+1 TO LEN z$)
1430 GO TO 1030
1440 PRINT ,,z$; FLASH 1;"?"
1450 GO TO 1030
1500 PRINT ,,TAB 4;"Hexadec. de REM a memoria"
1510 GO SUB 6000
1520 INPUT "Introduce num. de carac. hexad.";n
1530 IF n<1 OR n<>2*INT (n/2) THEN  BEEP .2,24: GO TO 1520
1540 LET l=5+PEEK 23635+256*PEEK 23636
1550 FOR i=l TO l+n-1 STEP 2
1560 LET j=PEEK i: LET k=PEEK (i+1)
1570 LET j=j-32*(j>96)+7*(j<65)-55
1580 LET k=k-32*(k>96)+7*(k<65)-55
1590 IF j<0 OR j>15 OR k<0 OR k>15 OR a>65535 THEN  GO TO 600
1600 POKE a,16*j+k
1610 LET a=a+1
1620 NEXT i
1630 GO TO 200
2000 PRINT ,,TAB 4;"Salida en hexadecimal"
2010 INPUT "Introduce direccion de comienzo para codigo    ";a:
     IF a<0 OR a<>INT a OR a>65536 THEN  BEEP .2,24: GO TO 2010
2020 LET fa=a
2030 IF fa-a=8*INT ((fa-a)/8) THEN  PRINT a;TAB 8;
2035 LET i=INT (PEEK a/16)
2040 PRINT CHR$ (i+55-7*(i<10));
2050 LET i=PEEK a-16*i
2060 PRINT CHR$ (i+55-7*(i<10));" ";
2070 IF INKEY$="s" OR INKEY$="S" THEN  GO TO 600
2080 LET a=a+1
2090 GO TO 2030
2200 INPUT "Introduce direccion de comienzo para codigo    ";a:
     IF a<16384 OR a<>INT a THEN  BEEP .2,24: GO TO 6000
2500 PRINT ,,TAB 11;"Hexadec. a REM"
2510 LET p=PEEK 23635+256*PEEK 23636
2520 LET l=PEEK (p+2)+256*PEEK (p+3)
2530 IF PEEK (p+4)<>234 THEN  BEEP .2,24:
     PRINT ,,"La primera sentencia no es REM": GO TO 600
2540 INPUT "Introduzca direccion comienzo   para codigo    ";a:
     IF a<0 OR a>65536 OR a<>INT a THEN  BEEP .2,24: GO TO 2540
2550 INPUT "Introduce num, de bytes a copiar";n: IF n<0 THEN
     BEEP .2,24: GO TO 2550
2560 IF n>l/2 THEN  BEEP .2,24: PRINT ,,"Solo hay sitio para  ";
     INT l/2;"bytes": GO TO 600
2570 LET p=p+5
2580 FOR i=a TO a+n-1
2590 LET j=INT (PEEK i/16)
2600 POKE p,j+55-7*(j<10)
2610 LET j=PEEK i-16*j
2620 POKE p+1,j+55-7*(j<10)
2630 LET p=p+2
2640 NEXT i
2650 GO TO 600
3000 PRINT ,,"Programa detenido"
3010 STOP
6000 INPUT "Introduzca direccion comienzo   para codigo    ";a:
     IF a<16384 OR a>65536 OR a<>INT a THEN  BEEP .2,24: GO TO 6000
6010 IF a>PEEK 23730+256*PEEK 23731 THEN  RETURN
```

```
6020 BEEP .2,24: INPUT "La direccion de comienzo es
     inferior al RAMTOP. Deseas con- tinuar (S/N)?";z$
6030 IF z$="S" OR z$="s" THEN  RETURN
6040 LET nr=599
6050 GO TO 9500
9000 FOR i=0 TO 20
9010 PRINT 65367-i,PEEK (65367-i)
9020 NEXT i
9030 STOP
9500 LET r=PEEK 23730+256*PEEK 23731-2
9510 IF PEEK r=2 THEN  LET r=r-3: GO TO 9510
9520 POKE r+1,nr-256*INT (nr/256)
9530 POKE r+2,INT (nr/256)
9540 RETURN
```

13. DESENSAMBLADOR

Usted probablemente ya sabe que el microprocesador Z80 que contiene el ZX SPECTRUM no entiende directamente palabras en BASIC como PRINT, IF, TAB, etc. En su lugar él sólo atiende a un lenguaje llamado CODIGO MAQUINA. Las instrucciones de la ROM de Sinclair, por ejemplo, están escritas en este código. El código máquina consiste en una secuencia de números enteros positivos comprendidos entre 0 y 255 (*). La función de un programa DESENSAMBLADOR es convertir una secuencia de estos números en una secuencia más comprensible de caracteres MNEMONICOS de ensamblador, que son alfanuméricos (un programa ENSAMBLADOR = ASSEMBLER tiene como objetivo complementario el convertir los MNEMONICOS en números enteros positivos).

Se pueden ver algunos ejemplos de los MNEMONICOS en el Apéndice A del Manual de Programación Basic para el ZX SPECTRUM (desde la página 183). La primera columna (CODIGO), incluye todos los números enteros positivos desde 0 a 255 y la cuarta columna (ZX ENSAMBLADOR) relaciona un MNEMONICO para cada uno de los números enteros positivos. Por ejemplo, el Mnemónicos "NOP" (se pronuncia "NO OP") es la contracción de "NO OPERATION" y su CODIGO MAQUINA es Ø, ya que Ø es la instrucción "NO HACER NADA". Es decir, el ZX80 ejecuta la instrucción "Ø" en código máquina NO HACIENDO NADA hasta que le llegue el turno a la siguiente instrucción.

Otro mnemónico simple es "INC b" (código máquina 4) que da orden el Z80 de incrementar el registro b en una unidad. O sea, si el registro b contiene 210 antes de ejecutar la instrucción, contendrá 211 después de ejecutarla.

El conjunto de instrucciones para el Z80 es complejo y no es el propósito de este libro descubrirlo tanto en forma como en estructura. Para ello recomendamos leer "HOW TO PROGRAM THE Z80" (cómo programar el Z80) de RODNEY ZAKS, editado por SYBEX (ISBN 0-89588-057-1) o "Z80 y 8080 ASSEMBLY LANGUAGE PROGRAMMING" (Lenguaje de programación ENSAMBLADOR para Z80 y 8080) de KATHE SPRACKLEN, editado por HAYDEN (ISBN 0-8104-5167-0).

El desensamblador debe interpretar correctamente todas las complejidades del conjunto de instrucciones del Z80. Pero existen cerca de 640 instrucciones diferentes para el Z80. Algunos como "NOP" e "INC b" se especifican tan sólo por un número natural, pero otros como "ld b, N" (código máquina 6) requieren 2, 3 o aún 4 números naturales. "ld b, N" significa acumular en el registro b la variable N definida en la siguiente instrucción. Así, la pareja de código máquina 6,15 hace que el Z80 almacene en el registro b el valor 15. Las variables se denominan mediante letras mayúsculas en la columna 4ª del Apéndice A del Manual de Programación Basic para el ZX SPECTRUM.

(*) Números naturales.

La otra forma en que las instrucciones requieren más de un número natural para ser completamente especificados es a través del uso de 4 números (203, 221, 237 y 253; CB, DD, ED y FD en hexadecimal) para formar un código ampliado. Los códigos ampliados para CD y ED se relacionan en las columnas 5 y 6 del Apéndice A. Así el par de números 237, 00 (CD 00 en hexadecimal) es el código para "ric b" que significa "rotación a la izquierda del registro b afectando al indicador (*)".

En otras palabras, cada uno de los 8 bits del registro b se trasladan un lugar a la izquierda dentro del byte, siendo trasladado el último de la izquierda al primer lugar de la derecha y al indicador (STACK POINTER).

Los códigos extendidos DD y FD no se relacionan en el Apéndice A debido a que son idénticos a los otros códigos con la única excepción de que los registros de 16 bits ix e iy son sustituidos por nl.

Por ejemplo, el código máquina 35 ("inc hl") ordena al Z80 que aumente en uno los 16 bits del registro hl. De igual manera, el par de instrucciones en código máquina 221, 35 (DD, 23 en hexadecimal) significa "inc ix" aumentar el registro ix, y el par 253, 35 (FD, 23 en hexadecimal) significa "inc iy" -aumentar el registro iy.

DESCRIPCION DEL PROGRAMA

La manera más evidente de escribir un programa DESENSAMBLADOR es construir una tabla "de búsqueda" conteniendo cada uno de los 640 MNEMONICOS "oficiales" (existen otras combinaciones más de códigos "NO oficiales", que se utilizan también). Sin embargo, un cálculo rápido nos muestra que no hay suficiente capacidad en un SPECTRUM CON 16K para almacenar cómodamente esta tabla y ADEMAS otro programa cualquiera. Suponiendo, por ejemplo, que cada mnemónico consta de 6 caracteres y un indicador numérico, que ocupa 5 bytes, la tabla completa (y el indicador) ocuparía tan sólo ella un mínimo de (6 + 5) x 640 = 7040 bytes. Si fuera posible utilizar indicadores enteros (ocupando 2 bytes cada uno), en lugar de la forma standard de SINCLAIR que ocupa 5 bytes, este espacio podría reducirse a (6 + 2) x 640 = 5120 bytes.

Un método más elegante y que yo he utilizado aquí es sacar partido de la simetría entre las partes del "Conjunto de Instrucciones" es construir algunos MNEMONICOS tomando como base sus componentes. Los códigos máquina entre 64 y 191 inclusive, por ejemplo, se refieren todos ellos a b, c, d, e, h, l (hl) y a registros en una secuencia repetida. Los números 64 a 127 agrupan todos ellos instrucciones "ld" y entre del 128 al 191 se pueden agrupar de 8 en 8 (add a, adc a, sub. etc.).

De forma similar los 2 códigos de byte que comienzan con el 203 (CB en hexadecimal) se relacionan en la columna 5 del Apéndice A y consisten en grupos de 8 en 8 acompañados por 3 grupos de 64. También muestran la misma secuencia repetitiva de referencias a los registros b, c, d, e, h, l, (hl) y a.

(*) NOTA DEL TRADUCTOR: Se traduce CARRY FLAG por INDICADOR (ver FLAGS en capítulo 25 del Manual de Programación Basic para el ZX SPECTRUM).

La Tabla TA1 muestra cómo analiza el desensamblador el código máquina.

Se utiliza un indicador w$ para distinguir entre los registros hl, ix e iy, y se cambia de forma adecuada si se encuentran los caracteres decimales 221 ó 225 (DD y FD en hexadecimal). Si se encuentra un código ampliado tipo ED (decimal 237) los mnemónicos se pueden buscar entre los caracteres de la cadena y$. Si se encuentra un código ampliado tipo CB (decimal 203) se construye el mnemónico entero a partir de sus componentes. Así una instrucción de un solo byte, o bien es leída de la cadena literal z$ o bien se construye a partir de sus componentes.

EL DESENSAMBLADOR EN DETALLE

El programa consiste en 2 secciones distintas. La primera sección (líneas 10 a 1340) almacena las Tablas de búsqueda (y$ y z$) y sus indicadores p y q. La segunda sección (líneas 3000 a 9010) es el DESENSAMBLADOR propiamente dicho. En su forma completa el programa ocupa casi las 16K de memoria y para comprimirlo es necesario cambiar el RAMTOP al más alto valor posible, lo cual se realiza introduciendo CLEAR 52767.

De hecho, una vez que los datos son almacenados mediante la ejecución del programa (RUN), las líneas 10 a 1340 pueden ser borradas en su totalidad teniendo en cuenta que las instrucciones RUN y CLEAR no van a ser utilizadas en adelante. El mejor método es ejecutar el programa entero mediante RUN, grabar una copia en cassette mediante SAVE, borrar las líneas 10 a 1340 y grabar una segunda copia mediante:

SAVE "DESENSAMB" LINE 3000

CODIGO MAQUINA (DECIMAL)		(HEXADECIMAL)	PROCEDIMIENTO
0	a 63	ØØ a 3F	Primeros 64 elementos de z$
64	a 191	4Ø a BF	Construidos en las líneas 3400 a 3510
192	a 255	CØ a FF	Ultimos 64 elementos de z$
203,0	a 255	CB, ØØ a FF	Construidos en las líneas 4500 a 4810
237,64	a 123	ED, 4Ø a 7B	Primeros 60 elementos de y$
237,160	a 187	ED, AØ a BB	Ultimos 28 elementos de y$
221,9	a 249	DD, Ø9 a F9	Ajuste de y$ en las líneas 4100 a 4150
253,9	a 249	FD, Ø9 a F9	Ajuste de z$ en las líneas 4100 a 4150
221,206,6	a 254	DD, C9, Ø6 a FE	Construido en las líneas 3400 a 3510 y ajustado mediante la subrutina 8000
253,203,6	a 254	FD, C9, Ø6 a FE	Construido en las líneas 3400 a 3510 y ajustado mediante la subrutina 8000

TABLA TA1. **El procedimiento utilizado por el DESENSAMBLADOR para analizar los diversos códigos máquina**

De esta forma, la copia original completa estará disponible en forma de fichero. Cuando se graba la segunda copia el programa comenzará automáticamente en la línea 3000 sin borrar las variables.

Los MNEMONICOS varían en longitud y así cada una de las tablas y$ y z$ requiere un indicador (q y p) para señalar el fin de cada mnemónico de forma ordenada. La primera de las secciones almacena los datos en los comandos DATA de las dos tablas. Por supuesto que no importa que se utilicen 11 sentencias DATA, como yo lo he hecho, o más, o menos, siempre que se mantenga su secuencia correcta.

La técnica utilizada para avanzar a través de las sentencias DATA en la segunda sección del programa (como por ejemplo entre las líneas 3400 a 3510) es de un valor notable. Esta es la parte que construye las instrucciones de la forma ld e, b y lo realiza leyendo (READ) el concepto correcto en las sentencias DATA en la línea 3400, almacenando el resultado y luego leyendo (READ) la misma sentencia una segunda vez. El número de conceptos leído se determina mediante los contadores situados respectivamente en las líneas 3440 y 3480, los cuales se calculan por el contenido del byte que está analizando.

La Tabla TA2 es un ejemplo de la salida del programa. Cada una de las direcciones y mnemónicos aparece ordenadamente.

```
1500    or e
1501    jr nz,-54
1503    ld a,h
1504    cp 1
1506    ret
1507    call 1511
1510    ret nc
1511    ld a,22
1513    dec a
1514    jr nz,-3
1516    and a
1517    inc b
1518    ret z
1519    ld a,127
1521    in a,(254)
1523    rra
1524    ret nc
1525    xor c
1526    and 32
1528    jr z,-13
1530    ld a,c
1531    cpl
```

TABLA TA2. Un ejemplo de la salida del programa DESENSAMBLADOR

PROGRAMA PA1. DESENSAMBLADOR

PROGRAMA PAI. DESENSAMBLADOR

```
 10 REM Esta seccion almacena los codigos de operacion en
    columnas 4 y 6 del Apendice A del Manual de Programacion
    Basic del ZX SPECTRUM
100 DIM p(128)
110 DIM z$(809)
120 DIM q(88)
130 DIM y$(392)
500 DATA "ld bc,NN","ld (bc),a","inc bc","inc b","dec b",
    "ld b,N","rlca","ex af,af","add hl,bc","ld a,(bc)","dec bc"
501 DATA "inc c","dec c","ld c,N","rrca","djnz DIS","ld de,NN",
    "ld (de),a","inc de","inc d","dec d","ld d,N","rla","jr DIS",
    "add hl,de","ld a,(de)","dec de","inc e","dec e","ld e,N",
    "rra","jr nz,DIS","ld hl,NN","ld (NN),hl","inc hl","inc h",
    "dec h"
502 DATA "ld h,N","daa","jr z,DIS","add hl,hl","ld hl,(NN)",
    "dec hl","inc l","dec l","ld l,N","cpl","jr nc,DIS",
    "ld sp,NN","ld (NN),a","inc sp","inc (hl)","dec (hl)",
    "ld (hl),N","scf","jr c,DIS","add hl,sp","ld a,(NN)"
503 DATA "dec sp","inc a","dec a","ld a,N","ccf"
510 DATA "ret nz","pop bc","jp nz,NN","jp NN","call nz,NN",
    "push bc","add a,N","rst 0","ret z","ret","jp z,NN","",
    "call z,NN","call NN","adc a,N","rst 8","ret nc","pop de",
    "jp nc,NN","out (N),a","call nc,NN","push de"
511 DATA "sub N","rst 16","ret c","exx","jp c,NN","in a,(N)",
    "call c,NN","","sbc a,N","rst 24","ret po","pop hl",
    "jp po,NN","ex (sp),hl","call po,NN","push hl","and N",
    "rst 32","ret pe","jp (hl)"
512 DATA "jp pe,NN","ex de,hl","call pe,NN","","xor N","rst 40",
    "ret p","pop af","jp p,NN","di","call p,NN","push af","or N",
    "rst 48","ret m","ld sp,hl","jp m,NN","ei","call m,NN","",
    "cp N","rst 56"
520 DATA "out (c),b","sbc hl,bc","ld (NN),bc","neg","retn",
    "im 0","ld i,a","in c,(c)","out (c),c","adc hl,bc",
    "ld bc,(NN)","","reti","","ld r,a","in d,(c)","out (c),d",
    "sbc hl,de","ld (NN),de"
521 DATA "","","im 1","ld a,i","in e,(c)","out (c),e",
    "adc hl,de","ld de,(NN)","","","im 2","ld a,r","in h,(c)",
    "out (c),h","sbc hl,hl","ld (NN),hl","","","","rrd",
    "in l,(c)","out (c),l","adc hl,hl","ld hl,(NN)"
522 DATA "","","","rld","in f,(c)","","sbc hl,sp","ld (NN),sp",
    "","","","","in a,(c)","out (c),a","adc hl,sp","ld sp,(NN)"
530 DATA "ldi","cpi","ini","outi","","","","","ldd","cpd","ind",
    "outd","","","","","ldir","cpir","inir","otir","","","","",
    "lddr","cpdr","indr","otdr"
1000 LET p(1)=3
1010 LET z$(1 TO 3)="nop"
1020 FOR i=1 TO 63
1030 READ i$: PRINT i;".......";i$
1040 LET p(i+1)=p(i)+LEN i$
1050 LET z$(p(i)+1 TO p(i+1))=i$
1060 NEXT i
1100 FOR i=192 TO 255
1110 READ i$: PRINT i;".......";i$
1120 LET p(i-127)=p(i-128)+LEN i$
1130 LET z$(p(i-128)+1 TO p(i-127))=i$
1140 NEXT i
```

```
1200 LET q(1)=8
1210 LET y$(1 TO 8)="in b,(c)"
1220 FOR i=65 TO 123
1230 READ i$: PRINT "ED  ";i;".......";i$
1240 LET q(i-63)=q(i-64)+LEN i$
1250 LET y$(q(i-64)+1 TO q(i-63))=i$
1260 NEXT i
1300 FOR i=160 TO 187
1310 READ i$: PRINT "ED  ";i;".......";i$
1320 LET q(i-99)=q(i-100)+LEN i$
1330 LET y$(q(i-100)+1 TO q(i-99))=i$
1340 NEXT i
2990 REM ******************************************************
     ******************************
2992 REM La siguiente seccion es el Desensmblador
3000 CLS
3010 PRINT TAB 10;"DESENSAMBLADOR"
3020 PRINT ,,"Introduce las direcciones de co-mienzo y de final"
3030 INPUT sa,sf
3040 IF sa<0 OR sf<sa OR sa<>INT sa OR sf<>INT sf THEN  BEEP .2,24
: GO TO 3030
3050 PRINT ,,"Deseas una copia en la impresora (S/N)?"
3060 INPUT c$: LET c$=CHR$ (CODE c$-32*(c$>"£")): IF c$<>"S"
     AND c$<>"N" THEN  BEEP .2,24: GO TO 3060
3100 LET i=sa
3110 LET j=PEEK i
3120 PRINT i;TAB 7;: IF c$="S" THEN  LPRINT i;TAB 7;
3130 LET w$="HL"
3140 GO TO 3200+1300*(j=203)+1800*(j=237)+2300*
     (j=221)+2500*(j=253)
3200 IF j<>0 THEN  GO TO 3230
3210 PRINT "nop": IF c$="S" THEN  LPRINT "nop"
3220 GO TO 9000
3230 IF j>63 AND j<192 THEN  GO TO 3400
3235 LET t=j-128*(j>191)
3240 FOR k=p(t)+1 TO p(t+1)
3250 IF z$(k TO k+1)="NN" THEN  GO TO 3800
3260 IF z$(k)="N" THEN  GO TO 3900
3270 IF z$(k TO k+2)="DIS" THEN  GO TO 4000
3280 IF z$(k TO k+1)="hl" AND w$<>"HL" THEN  GO TO 4100
3300 PRINT z$(k);: IF c$="S" THEN  LPRINT z$(k);
3310 NEXT k
3320 PRINT : IF c$="S" THEN  LPRINT
3330 GO TO 9000
3400 DATA "b","c","d","e","h","l","(hl)","a"
3405 IF j<>118 THEN  GO TO 3410
3406 PRINT "parar": IF c$="S" THEN  LPRINT "parar"
3407 GO TO 9000
3410 IF j>127 THEN  GO TO 3530
3420 LET i$="ld"
3430 RESTORE 3400
3440 FOR k=1 TO j/8-7
3450 READ j$
3460 NEXT k
3470 RESTORE 3400
3480 FOR k=1 TO j+1-8*INT (j/8)
3490 READ k$
3500 NEXT k
3510 PRINT i$;" ";j$;",";k$: IF c$="S" THEN
     LPRINT i$;" ";j$;",";k$
3520 GO TO 9000
3530 DATA "add","adc","sub","sbc","and","xor","or","cp"
```

```
3540 RESTORE 3530
3550 FOR k=1 TO j/8-15
3560 READ i$
3670 NEXT k
3680 IF j<144 OR (j>151 AND j<160) THEN  GO TO 3430
3690 RESTORE 3400
3700 FOR k=1 TO j+1-8*INT (j/8)
3710 READ j$
3720 NEXT k
3730 PRINT i$;" ";j$: IF c$="S" THEN  LPRINT i$;" ";j$
3740 GO TO 9000
3805 LET t=PEEK (i+1)+256*PEEK (i+2)
3810 PRINT t;: IF c$="S" THEN  LPRINT t;
3820 LET i=i+2
3830 LET k=k+1
3840 GO TO 3310
3900 LET t=PEEK (i+1)
3910 PRINT t;: IF c$="S" THEN  LPRINT t;
3920 LET i=i+1
3930 GO TO 3310
4000 LET t=PEEK (i+1)-256*(PEEK (i+1)>127)
4010 PRINT t;: IF c$="S" THEN  LPRINT t;
4020 LET i=i+1
4030 LET k=k+3
4040 GO TO 3310
4100 PRINT w$;: IF c$="S" THEN  LPRINT w$;
4110 LET k=k+1
4120 IF z$(k+1)<>")" OR j=233 THEN  GO TO 3310
4130 LET i=i+1
4140 PRINT "+";PEEK i;: IF c$="S" THEN  LPRINT "+";PEEK i;
4150 GO TO 3310
4500 LET i=i+1+(w$<>"HL")
4510 LET j=PEEK i
4520 IF j>63 THEN  GO TO 4700
4530 IF j<48 OR j>55 THEN  GO TO 4560
4540 PRINT "No hay tal codigo": IF c$="S" THEN
     LPRINT "No hay tal codigo"
4550 GO TO 9000
4560 DATA "rlc","rrc","rl","rr","sla","sra","","srl"
4570 RESTORE 4560
4580 FOR k=0 TO j/8
4590 READ i$
4600 NEXT k
4610 RESTORE 3400
4620 FOR k=0 TO j-8*INT (j/8)
4630 READ j$
4640 NEXT k
4645 IF j$="(hl)" AND w$<>"HL" THEN  GO SUB 8000
4650 PRINT i$;" ";j$: IF c$="S" THEN  LPRINT i$;" ";j$
4660 GO TO 9000
4700 DATA "bit","res","set"
4710 RESTORE 4700
4720 FOR k=1 TO j/64
4730 READ i$
4740 NEXT k
4750 RESTORE 3400
4760 FOR k=0 TO j-8*INT (j/8)
4770 READ j$
4780 IF j$="(hl)" AND w$<>"HL" THEN  GO SUB 8000
4790 NEXT k
4800 LET t=INT (j/8)-8*INT (j/64)
4810 PRINT i$;" ";t;",";j$: IF c$="S" THEN  LPRINT i$;" ";t;",";j$
4820 GO TO 9000
```

```
5000 LET i=i+1
5010 LET j=PEEK i
5020 IF j>63 AND j<124 THEN  GO TO 5100
5030 IF j>159 AND j<187 THEN  GO TO 5300
5040 GO TO 4540
5100 LET j=j-64
5105 IF j=0 THEN  GO TO 5400
5110 IF q(j)+1=q(j+1) THEN  GO TO 4540
5120 FOR k=q(j)+1 TO q(j+1)
5130 IF y$(k TO k+1)="NN" THEN  GO TO 5200
5150 PRINT y$(k);: IF c$="S" THEN  LPRINT y$(k);
5160 NEXT k
5165 PRINT : IF c$="S" THEN  LPRINT
5170 GO TO 9000
5200 LET t=PEEK (i+1)+256*PEEK (i+2)
5210 PRINT t;: IF c$="S" THEN  LPRINT t;
5220 LET i=i+2
5230 LET k=k+1
5240 GO TO 5160
5300 LET j=j-159
5310 IF p(j)+1=p(j+1) THEN  GO TO 4540
5320 PRINT y$(p(j)+1 TO p(j+1)): IF c$="S" THEN
     LPRINT y$(p(j)+1 TO p(j+1))
5330 GO TO 9000
5400 PRINT y$(1 TO 8): IF c$="S" THEN  LPRINT y$(1 TO 8)
5410 GO TO 9000
5500 LET w$="ix"
5510 LET i=i+1
5520 LET j=PEEK i
5530 GO TO 3140
5700 LET w$="iy"
5710 GO TO 5510
8000 IF PEEK (i-1)=253 OR PEEK (i-1)=221 THEN  GO TO 8030
8010 LET j$="("+w$+"+"+STR$ PEEK (i-1)+")"
8020 RETURN
9000 LET i=i+1
9010 IF i<=sf THEN  GO TO 3110
```

14. EDITOR DE CODIGO MAQUINA

Con cualquier lenguaje de programación y para cualquier computador parece siempre que nos encontramos en el caso de existir multitud de tareas que el usuario desea que la máquina pueda realizar pero que no pueden ser convenientemente escritas en el lenguaje disponible. El ZX SPECTRUM no se escapa a esta consideración.

Examinemos, por ejemplo, el problema de trasladar el contenido de la pantalla a la parte más alta de la memoria RAM. La salida por pantalla y sus características ocupan 6912 bytes y así es necesario mover el RAMTOP hacia abajo a 32768 - 6912 = 25856 en una máquina de 16K (65536 - 6912 = 58624 en una máquina de 48K). El siguiente programa de BASIC, muy sencillo, introducirá la salida por pantalla en un área reservada pero tardará bastante tiempo en hacerlo -cerca de 70 segundos-.

```
10 FOR i = Ø TO 6911
20 POKE 25856 + i, PEEK (16384 + i)
30 NEXT i
```

la razón de que tarde tanto tiempo se debe a que el SPECTRUM emplea la mayor parte de su tiempo decodificando comandos, convirtiendo a otros números las parejas de números naturales que el Z80 comprende y los 5 bytes decimales en los cuales se almacena el contador del bucle y, finalmente, ejecutar las operaciones aritméticas indicadas por los 5 bytes.

Las etapas de este proceso son las siguientes:

1. Añadir i a 16384
2. Convertir el resultado a la forma de 2 bytes
3. Recuperar el contenido de la dirección PEEK
4. Añadir i a 25856
5. Convertir el resultado a la forma de 2 bytes
6. Almacenar el valor recuperado en la dirección POKE
7. Añadir UNO al valor de i y almacenar el resultado
8. Restar (i - 6911). Si el resultado es positivo o cero, GOTO 1.

Cada vez que se realiza una pasada a través del bucle el SPECTRUM debe decodificar cada comando nuevamente, puesto que no retiene en la memoria las operaciones previas. Es fácil darse cuenta que el computador pierde cerca del 90% preparándose para ejecutar los comandos más bien que ejecutándolos. Tampoco es ninguna sorpresa el que una subrutina en código máquina realice la operación de almacenar la salida por pantalla en un sitio de la memoria arriba del RAMTOP de forma casi instantánea. Un ejemplo de esta subrutina se acompaña al programa HEX LOADER.

Desgraciadamente, escribir en código máquina es una tarea difícil y no es juicioso el intentar hacerlo sin la ayuda de algo de software. Indudablemente, el mejor programa para escribir en código máquina (como opuesto a un lenguaje compilador) es un ensamblador, pero un ensamblador debe también escribirse en código máquina y no es demasiado cómodo publicar largas listas de programas en código máquina. Así, he incluido en este libro un EDITOR DE CODIGO MAQUINA en su lugar.

El Editor proporciona todas las posibilidades más importantes de un ensamblador excepto para la conversión de los MNEMONICOS en hexadecimal.

Utilizando el Editor puede escribirse CODIGO MAQUINA en caracteres hexadecimales en un área reservada de memoria, añadiendo y borrando partes de código, asignando y nombrando variables de 8 bits y 16 bits, nombrando segmentos de código, llamando o saltando a aquellos segmentos mediante su nombre y almacenando el CODIGO absoluto en la parte más alta del RAM en la forma de caracteres. El Editor se escribe completamente un BASIC y de esta forma las posibilidades de añadir, borrar y almacenar tardan un tiempo razonable -normalmente entre 5 y 20 segundos- en ejecutarse.

UN ESQUEMA DE PROGRAMA

El programa es controlado desde un MENU de opciones (línea 2470) ilustrado en la figura FM1. La línea 2490 llama a las distintas subrutinas que dependen de la opción elegida dentro del MENU.

```
Editor de codigo maquina

          Menu

Anadir REM              A
Cargar una rutina       L
Nombrar un segmento     N
Sacar una rutina        U
Insertar caract. hex    I
Borrar caract. hex      D
Revisar caract. hex     R
Escribir caract. hex    W
Parar                   S

 Espacio para caract. hex desde
23760 hasta 23971
```

FIGURA FM1. **MENU de opciones del EDITOR DE CODIGO MAQUINA**

AÑADIR UNA SENTENCIA REM

Introducir A

El código hexadecimal que introduce el usuario mediante teclado se almacena en la línea 1. El programa toma nota del número de posiciones disponible en la sentencia REM y el resultado aparece en pantalla en la parte inferior del MENU de opciones. Si se necesita más espacio entonces el usuario debe parar el programa e introducir una sentencia REM de longitud adecuada en la línea 2, ejecutar (RUN) el programa y después seleccionar esta opción (A). La nueva sentencia REM se incorpora a la antigua y la cantidad de espacio disponible para el código hexadecimal se incrementa en consecuencia.

La subrutina que controla esta opción se encuentra en las líneas 2430 a 2460 y 2600 a 2660. Cuando se ejecuta el programa (RUN) la longitud de la primera línea de programa se almacena en 1 y se realiza una verificación para comprobar que la primera sentencia es una REM (por ejemplo que los 5 bytes en el área del programa contienen 234). Cuando la opción de "AÑADIR" se selecciona se realiza otra comprobación para asegurarse que la segunda sentencia es una REM, calculándose un nuevo valor de 1, el cual se introduce mediante POKE en el indicador de longitud de línea (como la vez anterior).

ESCRIBIR CARACTERES HEXADECIMALES — Introducir W

La subrutina para esta opción se encuentra entre las líneas 2680 y 2850. El usuario debe introducir la dirección (dentro del espacio hexadecimal disponible) dentro de la cual él desea almacenar el código, hecho lo cual el programa le invita a introducir el código en grupos. El programa señala el final de cada uno de los grupos mediante la adición de 128 al código del carácter, de tal forma que la opción "REVISAR" pueda romper este código en los mismos grupos. El programa copia lo introducido mediante el teclado en la pantalla junto con las direcciones donde se almacenan los códigos.

El usuario siempre puede pulsar la tecla H (para parar) como código final de un programa, puesto que la subrutina que realiza el almacenamiento para cuando se encuentra una H. Para terminar la introducción de caracteres en medio de un programa pulse la letra T.

El siguiente ejemplo muestra como una subrutina para imprimir el indicador (STACK POINTER) puede introducirse en el espacio para caracteres hexadecimales.

CARACTERES INTRODUCIDOS	COMENTARIO
W	Se selecciona la opción ESCRIBIR (caract. HEX)
23760	Dirección de comienzo del espacio hexadecimal
210000	LD HL,
39	ADD HL, SP
44	LD B, H
4D	LD C, L
C9	RET
H	Señal de «PARADA» para el almacenamiento
T	se ha terminado la introducción de caracteres.

REVISAR CARACTERES HEXADECIMALES — Introducir R

Esta subrutina está entre las líneas 2900 y 3120. La dirección de cada grupo de código se escribe acompañada de pares de códigos y de forma ordenada.

INTRODUCIR Y BORRAR CARACTERES HEXADECIMALES

Introducir I ó D

Los caracteres hexadecimales pueden ser introducidos o borrados en cualquier punto del espacio hexadecimal. En ambos casos se ha de tener cuidado de introducir el número de CARACTERES que han de ser introducidos o borrados y **NO** EL NUMERO DE BYTES del código compilado. Por ejemplo, si el código hexadecimal para "inc hl" es 23 entonces se ha de introducir 2 porque el número de caracteres es 2 aunque el código compilado ocupe sólo un byte.

La subrutina de borrado se encuentra entre las líneas 2340 y 2400 y la subrutina de inserción entre las líneas 2160 y 2220. Después de crear el suficiente número de espacios requeridos, la subrutina de inserción salta a la subrutina de escritura, de tal forma que el usuario puede crear el espacio y escribir encima el código siguiente con la misma opción.

NOMBRAR UN SEGMENTO

Introducir N

Esta subrutina (entre las líneas 3130 y 3300) introduce un NOMBRE en la localización elegida por el usuario. El nombre es precedido y acompañado de un PUNTO cuando es introducido en una sentencia REM, haciéndolo más fácil de identificar tanto para el usuario como para el programa.

Los dos bytes que acompañan al nombre se utilizan para almacenar un punto de entrada para el segmento relativo a la posición del nombre.

Este método permite al usuario situar una subrutina al comienzo de un segmento y pasarla por alto cuando el segmento es llamado primero.

LLAMANDO SEGMENTOS Y NOMBRANDO VARIABLES

La opción de escribir (W) necesita códigos hexadecimales como INPUT pero si se introduce un nombre de segmento (precedido y acompañado de un GUION) entonces también dicho nombre será introducido en la sentencia REM. De un modo similar pueden ser introducidas variables de 8 bits y 16 bits, siendo sólo aceptadas en las formas S1, S2, S3, etc. (variables de 8 bits) y L1, L2, L3, etc (variables de 16 bits).

ALMACENANDO UNA SUBRUTINA **Introducir L**

Esta es la subrutina más complicada del programa. Primero se pregunta al usuario por la dirección en que va a ser almacenada la versión de código absoluto. La respuesta normal podría ser: en un área por encima del RAMTOP. Si la respuesta del usuario es una dirección que se encuentra por debajo del indicador del STKEND (p.e.: en el área ocupada por el propio EDITOR DE CODIGO MAQUINA) entonces el programa pide una confirmación de dicha dirección.

La subrutina realiza dos pasadas a través del código hexadecimal de las sentencias REM. En la primera pasada cuenta el número de recorridos del bucle las variables de 8 bits y su cantidad, las variables de 16 bits y almacena los nombres de todos los segmentos del programa en y$. Antes de comenzar la segunda pasada se asegura suficiente espacio para las variables al principio del área de código absoluto, calculando los puntos absolutos introducidos en todos los segmentos.

En la segunda pasada, el programa compila el código en su sitio sustituyendo la dirección absoluta de variables y segmentos de forma correcta.

LLAMAR UNA SUBRUTINA **Introducir U**

Esta subrutina puede utilizarse para comprobar el código absoluto.

Imprime el Código hexadecimal para cada byte entre los límites especificado por el usuario.

UN EJEMPLO

Para una salida por pantalla sugerente introduzca lo siguiente:

CARACTERES INTRODUCIDOS	COMENTARIO
9500 POKE 31000, Ø	Introduce (POKE) la dirección del
9510 POKE 31001, 64	principio de la salida por pantalla
9520 POKE 31002, 255	y del final de los atributos en
9530 POKE 31003, 91	31000 - 31003
9540 RANDOMIZE USR 31004	Llamar a la subrutina en código máquina
9550 GOTO 9540	
RUN	Ejecuta el programa
W	Escribe una subrutina en
23760	23760
2AL1	LD HL, (L1)
EB	EX, DE, HL
2AL2	LD HL, (L2)
A7	AND A
ED 52	SBC HL, DE
44	LD B, H
4D	LD C, L
2AL1	LD HL, (L1)
3A785C -NEXT-	LD A, (23672)
77	LD (HL), A
23	INC HL
ØB	DEB BC
78	LD A, B
BC	OR C
FE	CP,
C2 -NEXT-	JP NZ NEXT
C9	RET
H	
T	
N	Nombrando una subrutina
23784	en 23784
NEXT	Nombre de una subrutina
Ø	Punto de entrada relativo
L	Almacenando una subrutina
31000	en 31000
S	Deteniendo el Editor de Código Máquina
RUN 9500	EJECUTANDO el código básico

CODIGOS DE ERROR

1. Intento de introducir o borrar un código fuera del espacio hexadecimal.
2. La primera línea no es una sentencia REM (error fatal).
3. El comando que se añade a la línea 1 no es REM (error fatal).
4. Intento de comenzar el código hexadecimal fuera del espacio hexadecimal.
5. Intento de continuar el código hexadecimal más allá del espacio hexadecimal.
6. Intento de revisar el código hexadecimal fuera del espacio hexadecimal.
7. Intento de añadir un nombre fuera del espacio hexadecimal.

PROGRAMA PM1. EL EDITOR DE CODIGO MAQUINA

```
   1 REM 1111111111111111111111111111111111111111111111111111111111111111
         1111111111111111111111111111111111111111111111111111111111111111
         1111111111111111111111111111111111111111111111111111111111111111
         111111111111
  20 LET t6=6
  30 LET t24=27
1000 GO TO 2430
1010 LET i=i+1
1030 FOR j=1 TO LEN y$
1040 IF PEEK i=CODE y$(j TO ) THEN  GO TO 1080
1050 NEXT j
1060 LET b=-1
1070 RETURN
1080 FOR p=1 TO LEN y$-j-2
1090 IF PEEK (i+p)=CODE "." OR PEEK (i+p)=173 THEN  GO TO 1130
1100 IF PEEK (i+p)<>CODE y$(j+p TO ) THEN  GO TO 1050
1120 NEXT p
1130 LET b=CODE y$(j+p TO )+256*CODE y$(j+p+1 TO )
1140 LET i=i+p+1-2*(PEEK (i+p)=173)
1150 RETURN
1160 GO SUB 1010
1170 IF b>32767 AND b<65536 THEN  GO TO 1200
1180 PRINT ,,"Error 11"
1190 STOP
1200 LET y$(j+p TO j+p+1)=CHR$ (b+k-256*INT ((b+k)/256))+
     CHR$ (INT ((b+k)/256)-128)
1210 PRINT k;"  ";y$(j TO j+p-1)
1220 GO TO 2070
1230 LET b=lo+2*(PEEK (i+1)-49-7*(PEEK (i+1)>57)-121*
     (PEEK (i+1)>127))
1260 POKE k,b-256*INT (b/256)
1270 POKE k+1,INT (b/256)
1280 LET k=k+2
1290 GO TO 2070
1300 LET b=s+PEEK (i+1)-49-7*(PEEK (i+1)>57)-121*(PEEK (i+1)>127)
1310 GO TO 1240
1320 GO SUB 1010
1330 GO TO 1240
1370 LET i=5+PEEK 23635+256*PEEK 23636
1380 LET k=i
1390 PRINT TAB 7;"Cargar codigo maquina",,,
     "Introducir direccion para codigo"
```

```
1400 INPUT j: IF j>16384 AND j<PEEK 23653+256*PEEK 23654 THEN
     PRINT ,,"Esta parte de RAM esta ocupada. Quieres seguir
     (S o N)?": INPUT z$: IF z$="N" OR z$="n" THEN
     PRINT "OK. Prueba otra vez": GO TO 1400
1480 LET lo=0
1490 LET s=0
1500 GO SUB 6000
1510 IF p=CODE "-" OR p=CODE "." THEN  GO TO 1540
1515 IF p>CODE "F" THEN  GO TO 1600
1520 LET i=i+1
1530 GO TO 1500
1540 LET i=i+1
1550 GO SUB 6000
1560 IF p=CODE "-" THEN  GO TO 1520
1570 IF p<>CODE "." THEN  GO TO 1540
1580 LET i=i+3
1590 GO TO 1500
1600 IF p=CODE "H" THEN  GO TO 1750
1602 IF p=CODE "H" THEN  GO TO 1750
1610 LET q=PEEK (i+1)-48-128*INT (PEEK (i+1)/128)
1650 IF p=CODE "L" THEN  GO TO 1690
1660 IF p=CODE "S" THEN  GO TO 1720
1690 IF q>lo THEN  LET lo=q
1700 LET i=i+2
1710 GO TO 1500
1720 IF q>s THEN  LET s=q
1730 LET i=i+2
1740 GO TO 1500
1750 PRINT ,,lo;" variables largas",s;" variables cortas"
1760 LET i=k
1770 LET k=j+2*lo+s
1780 LET ks=k
1790 LET s=2*lo+j
1800 LET lo=j
1810 PRINT " variables largas en ";lo,"variables cortas en ";s,
     "programa en c/m empieza en ";k
1830 LET m=i
1840 LET m=i: LET y$=""
1860 GO SUB 6000
1862 IF p=CODE "." THEN  GO TO 1900
1870 IF p=CODE "H" THEN  GO TO 1970
1880 LET i=i+1
1890 GO TO 1860
1900 LET i=i+1
1910 IF PEEK i=CODE "." THEN  GO TO 1940
1920 LET y$=y$+CHR$ PEEK i
1930 GO TO 1900
1940 LET y$=y$+CHR$ PEEK (i+1)+CHR$ PEEK (i+2)
1950 LET i=i+3
1960 GO TO 1860
1970 LET i=m
1980 GO SUB 6000: IF p=CODE "H" THEN
     PRINT ,,"Final en ";k-1: RETURN
1990 IF p=CODE "." THEN  GO TO 1160
2000 IF p=CODE "L" THEN  GO TO 1230
2010 IF p=CODE "S" THEN  GO TO 1300
2020 IF p=CODE "-" THEN  GO TO 1320
2050 POKE k,16*(PEEK i-7*(PEEK i>57))-816+PEEK (i+1)-7*
     (PEEK (i+1)>57)-121*(PEEK (i+1)>127)-7*(PEEK (i+1)>185)
2060 LET k=k+1
2070 LET i=i+2
```

```
2080 GO TO 1980
2090 PRINT TAB 10;"Sacar codigo",,,
     "Introduce las direcciones de co-mienzo y final"
2100 INPUT ks,kf
2110 FOR i=ks TO kf
2130 LET p=INT (PEEK i/16): LET q=PEEK i-16*INT (PEEK i/16):
     PRINT CHR$ (48+p+7*(p>9));CHR$ (48+q+7*(q>9));"  ";
2140 NEXT i
2150 RETURN
2160 PRINT ,,TAB 10;"Introducir codigo",,,
     "Direccion en la que se introducey num. de conceptos"
2170 INPUT i,j
2180 IF i<prog OR j<0 OR i+j>prog+1 THEN  BEEP .2,0:
     PRINT "Error 1": PAUSE 0: GO TO 2170
2190 FOR k=prog+1-1 TO i+j STEP -1
2200 POKE k,PEEK (k-j)
2210 NEXT k
2220 GO TO 2700
2340 PRINT ,,TAB 10;"Borrar codigo",,,
     "Direccion de comienzo de borradoy num de caracteres hex"
2350 INPUT i,j
2360 IF i<prog OR j<0 OR i>prog+1 THEN  BEEP .2,0:
     PRINT "Error 1": PAUSE 0: GO TO 2350
2370 FOR k=i TO prog+1-j-1
2380 POKE k,PEEK (k+j)
2390 NEXT k
2400 RETURN
2430 LET prog=5+PEEK 23635+256*PEEK 23636
2440 LET l=PEEK (prog-3)+256*PEEK (prog-2)-2
2450 CLS : PRINT TAB 6;"Editor de codigo maquina"
2460 IF PEEK (prog-1)<>234 THEN  BEEP .2,0: PRINT "Error 2": STOP
2470 PRINT ,,,,TAB 14;"Menu",,,TAB t6;"Anadir REM";TAB t24;"A";
     TAB t6;"Cargar una rutina";TAB t24;"L";TAB t6;
     "Nombrar un segmento";TAB t24;"N";TAB t6;"Sacar una rutina";
     TAB t24;"U";TAB t6;"Insertar caract. hex";TAB t24;"I";TAB t6;
     "Borrar caract. hex";TAB t24;"D";TAB t6;
     "Revisar caract. hex";TAB t24;"R";TAB t6;
     "Escribir caract. hex";TAB t24;"W";TAB t6;"Parar";
     TAB t24;"S",,,"  Espacio para caract. hex desde ";prog;
     " hasta ";prog+l,,,
2480 INPUT z$: IF CODE z$>90 THEN  LET z$=CHR$ (CODE z$-32)
2490 GO SUB 2400+150*(z$="S")+200*(z$="A")-1030*(z$="L")+730*
     (z$="N")-310*(z$="U")-240*(z$="I")-60*(z$="D")+500*
     (z$="R")+280*(z$="W")
2500 PRINT TAB 0,,"Pulsa una tecla para continuar": PAUSE 0
2510 GO TO 2430
2550 PRINT ,,"programa detenido"
2560 STOP
2610 IF PEEK (prog+l+5)<>234 THEN  BEEP .2,0:
     PRINT "Error 3": STOP
2620 LET l=l+PEEK (prog+l+3)+256*PEEK (prog+l+4)+6
2630 POKE prog-3,l-256*INT (l/256)
2640 POKE prog-2,INT (l/256)
2650 LET l=l-2
2660 RETURN
2680 PRINT ,,TAB 9;"Escribir codigo hex.",,,
     "Introducir direccion de comienzo"
2690 INPUT i: IF i<prog OR i>prog+1 THEN  BEEP .2,0:
     PRINT "Error 4": PAUSE 0: GO TO 2690
2700 PRINT ,,"Introduce programa hexadecimal
     (pulsa t para salir)"
```

```
2710 INPUT z$: IF z$="t" OR z$="T" THEN  RETURN
2720 LET f=0
2730 PRINT i;"  ";
2740 FOR j=1 TO LEN z$
2750 IF i>prog+1 THEN  BEEP .2,0: PRINT "Error 5": RETURN
2760 LET z$(j)=CHR$ (CODE z$(j)-32*(CODE z$(j)>96))
2770 POKE i,CODE z$(j)
2780 PRINT z$(j);
2790 LET f=f=1 OR z$(j)="-" OR z$(J)="."
2800 IF f=0 AND j=2*INT (j/2) THEN  PRINT " ";
2810 LET i=i+1
2820 NEXT j
2830 PRINT
2840 POKE i-1,PEEK (i-1)+128
2850 GO TO 2710
2900 PRINT ,,TAB 5;"Revisar codigo hexadecimal",,,
     "Introducir dir. de comienzo"
2910 INPUT i
2920 IF i<prog OR i>prog+1 THEN  BEEP .2,0:
     PRINT "Error 6": GO TO 2910
2940 LET j=0
2950 PRINT i;"  ";
2960 GO TO 2970+90*(PEEK i=CODE ".")+80*(PEEK i=CODE "-")
2970 PRINT CHR$ (PEEK i-128*INT (PEEK i/128));
2975 IF PEEK i=200 THEN  PRINT TAB 0;"Fin del codigo": RETURN
2980 LET i=i+1
2990 LET j=j+1: IF j=2*INT (j/2) THEN  PRINT ;" ";
3010 IF PEEK (i-1)<128 THEN  GO TO 2960
3020 PRINT
3030 GO TO 2930
3050 PRINT "-";
3060 LET i=i+1
3070 IF PEEK i=173 THEN  GO TO 2970
3080 PRINT CHR$ PEEK i;
3090 IF PEEK (i+1)<>CODE "." THEN  GO TO 3060
3100 PRINT " entrada rel. en ";PEEK (i+2)+256*(PEEK (i+3)-128)
3110 LET i=i+4
3120 GO TO 2930
3130 PRINT TAB 9;"Nombrar un segmento",,,
     "Introducir direccion del segmen."
3140 INPUT i: IF i<prog OR i>prog+1 THEN  BEEP .2,0:
     PRINT "Error 7": GO TO 3140
3150 PRINT ,,"Introducir nombre"
3160 INPUT z$
3170 LET z$="."+z$+"."
3180 LET j=LEN z$+2
3190 FOR k=prog+1 TO i+j STEP -1
3200 POKE k,PEEK (k-j)
3210 NEXT k
3220 FOR j=1 TO LEN z$
3225 IF CODE z$(j)>96 THEN  LET z$(j)=CHR$ (CODE z$(j)-32)
3230 POKE i+j-1,CODE z$(j)
3240 NEXT j
3250 LET i=i+LEN z$
3260 PRINT ,,"Introduce offset del punto de entrada"
3270 INPUT j
3280 POKE i,j-256*INT (j/256)
3290 POKE i+1,128+INT (j/256)
3300 RETURN
6000 LET p=PEEK i-128*INT (PEEK i/128)
6010 RETURN
9000 FOR i=0 TO 255
9010 POKE 22528+i,i
9020 NEXT i
9030 PRINT USR 32600
```

15. GRAFICOS

Cuando comencé a trabajar en este programa lo ví con un sentido de generación de formas que podían ser grabadas en cassette utilizando el comando SCREEN $ y cargados posteriormente en otros programas.

Sin embargo, pronto comprobé que, con algunos cambios de tipo menor podía escribir un programa que permitiera al usuario general, almacenar y utilizar sus propios CARACTERES GRAFICOS.

PRINCIPIOS OPERATIVOS

Veintiuno (21) de los 256 códigos de carácter en el GRUPO de caracteres del SPECTRUM (números 144 al 164 inclusive) se encuentran a disposición para que el usuario defina sus propios caracteres gráficos. La tabla que define la forma de los caracteres normalmente se encuentra en la parte más alta del RAM, justo sobre el indicador del RAMTOP (tal como se indica en la página 179 del Manual de Programación BASIC del ZX SPECTRUM).

Cuando se conecta a la red el ZX SPECTRUM en los gráficos definidos por el usuario hay una copia de las letras mayúsculas A a U y la variable del sistema UDG en 23675 y 23676 está apuntando a la cota inferior del área de gráficos en 32600. Se puede acceder a los caracteres gráficos presionando CAPS SHIFT 9, seguidos de la letra deseada y a continuación CAPS SHIFT 9 de nuevo.

Todos los caracteres del ZX SPECTRUM se construyen mediante una retícula de 8 x 8 pixels, pudiendo estar cada uno de los pixels iluminados o no.

Esto permite al bloque de pixels que forman un carácter ser almacenados en 8 bytes con la condición de que cada uno de los 8 bits de cada byte tenga un cero o un uno si se desea que el pixel correspondiente esté iluminado o no. Así las condiciones deseadas para los 8 pixels de la línea superior de la retícula se almacenan en el primer byte, la 2ª línea se almacenan en el segundo byte y así los demás.

Conforme a este método la forma de los 21 caracteres diferentes (letras mayúsculas de la A a la U) puede ser almacenada en una tabla de 21 x 8 = 168 bytes. No es sorprendente pues ver que el indicador UDG se encuentre en 32600, es decir 168 bytes por debajo de la parte más alta del RAM (RAMTOP), para una máquina de 16K.

Si se introduce un nuevo valor para un carácter dentro de la variable del sistema UDG el área de memoria (a la que se accede cuando se construyen gráficos definidos por el usuario) cambiará así como los caracteres construidos en ella. Este efecto se ilustra por la subrutina que aparece a continuación, la cual imprime una línea de caracteres gráficos, cambia UDG e imprime una segunda línea, etc.

```
10 FOR i=1 TO 21
20 PRINT AT i,0;"ABCDEFGHIJKLMNOPQRSTUABCDEFGHIJK"
30 LET J=RND*32768
40 POKE 23675,INT (J-256*INT (J/256))
50 POKE 23676,INT (J/256)
60 NEXT I
70 GO TO 10
```

Nota:Las letras son los caracteres definibles

PROGRAMA PG1. **Este programa imprime caracteres gráficos fantásticos**

La línea de caracteres en el comando PRINT de la línea 20 debe ser precedido y terminado por CAPS SHIFT 9 cuando se introduzca mediante teclado.

SOBRE EL PROGRAMA

Cuando el programa se ejecuta (RUN) el RAMTOP se baja a 31087 para disponer de espacio suficiente para 10 grupos de caracteres definidos por el usuario de 21 caracteres cada uno. A continuación el programa accede a una subrutina de inicialización (líneas 500 a 900), la cual almacena 32 caracteres para el diseño de su forma en el primero y parte del segundo almacén de caracteres. Los datos para estos caracteres se leen en las sentencias DATA y algunos de los caracteres resultantes pueden verse en las sentencias PRINT de las líneas 760, 1060 y 2610. Se puede también acceder a esta subrutina de inicialización desde el MENU principal, mediante la opción I.

El MENU principal se muestra en la figura FG1 y aparece en pantalla una vez que se ha llevado a cabo la inicialización. Su parte superior de la pantalla es una retícula de 8 por 8 que se utiliza para construir interactivamente nuevos caracteres, obteniendo caracteres tanto de la forma ordinaria como de los ya definidos por el usuario, pudiéndolos intercambiar. Esta última característica permite al usuario copiar los caracteres antiguos y alterarlos en algo o en todo para producir unos nuevos.

Cuando se selecciona la opción "AÑADIR", el programa pregunta al usuario si desea BORRAR la SALIDA DE GRAFICOS (GRAPHIX DISPLAY) y después se le invita a introducir pares de coordenadas para crear o ampliar el carácter gráfico. Después de cada par de coordenadas el programa comprueba las características de localización de impresión adecuadas, si el carácter gráfico número 8 está en blanco y pone a cero los valores de INK y PAPER. Este procedimiento permite que los caracteres se impriman sobre los antiguos, pero sin borrarlos, como sería el caso si hubieramos utilizado el comando OVER.

```
GRAFICOS

12345678
8 ▌▌▌▌▌▌▌▌ 8
7 ▌▌▌▌▌▌▌▌ 7
6 ▌▌▌▌▌▌▌▌ 6
5 ▌▌▌▌▌▌▌▌ 5
4 ▌▌▌▌▌▌▌▌ 4
3 ▌▌▌▌▌▌▌▌ 3
2 ▌▌▌▌▌▌▌▌ 2
1 ▌▌▌▌▌▌▌▌ 1
12345678

Anadir caracteres                A
Salida caracteres en pantalla    D
Volcado desde mem. a caracter    M
Copia de caracter a memoria      C
Construir Pant.                  B
Grabar prog.                     S
Parar prog.                      H
```

FIGURA FG1. **El MENU de opciones de GRAFICOS**

La opción D, "salida de todos los caracteres", borra la pantalla e imprime todos los gráficos definidos por el usuario. La opción M, "memoria a salida de caracteres por pantalla", copia un carácter tanto si es uno de los 10 definidos por el usuario como si es uno de los standard (número CERO) en la SALIDA GRAFICA y la opción C, "caracteres a memoria" realiza la función contraria.

El programa, las variables y el área de caracteres gráficos puede ser grabado en cassette utilizando la opción "GRABAR el programa en cinta", pero antes de cargarlo (de cinta) el usuario debe introducir por teclado para borrar el RAMTOP:

CLEAR 31087

CONSTRUYENDO UNA SALIDA POR PANTALLA **(Introducir B)**

Habiendo diseñado caracteres a nuestra conveniencia, almacenándolos en la memoria y, quizás, grabándolos en cinta utilizando el MENU principal, el usuario puede seleccionar la opción "CONSTRUIR UNA SALIDA GRAFICA" para diseñar una salida por pantalla de su elección. Las 20 líneas por 32 columnas de la pantalla están disponibles para el usuario, aunque la 20 y 21 son utilizadas para mostrar los caracteres gráficos. La salida puede construirse a partir de todos los caracteres de los 10 grupos definidos por el usuario o a partir de los caracteres ordinarios. Una salida puede ser grabada en cassette y ser utilizada en un programa posterior. De igual forma, una salida de otro programa distinto puede ser copiada en la pantalla.

Cuando se selecciona la "B" aparecen varias opciones en pantalla, tal como se ilustra en la figura FG2. Sólo se permiten las letras mayúsculas A a U excepto cuando se usan los caracteres ordinarios. Una letra de la "A" a la "U" seguida de una pareja de coordenadas hace que aparezca en la pantalla el carácter correspondiente dentro de los definidos por el usuario. Las letras pueden ser repetidas tan frecuentemente como se desee para construir la forma requerida, dibujar un mapa, etc. Otro grupo de caracteres puede ser seleccionado sin necesidad de alterar el contenido de la pantalla mediante la letra "Y". El nuevo grupo aparece en la parte inferior de la pantalla. El grupo ordinario de caracteres se selecciona de forma similar mediante la letra "V".

Las letras "W" y "X" se utilizan respectivamente para GRABAR en cinta y CARGAR (de cinta) la pantalla. "Z" hace volver al MENU principal.

```
Construir PANTALLA

Comandos :

A-U - dibuja A-U  grupo actual

V   - car. standard(salir=SPACE)

W   - graba pantalla en cinta

X   - carga pantalla de cinta

Y   - cambia otro grupo caract.

Z   - menu

Pulsa tecla para seguir
```

FIGURA FG2. **Las varias opciones disponibles cuando se construye una salida por pantalla**

SPECTRUM CON 48K

El programa funciona tal y como está escrito para la máquina de 48K. El espacio extra disponible sobre el RAMTOP puede utilizarse para almacenar copias de diversas salidas por pantalla. De las 2 subrutinas que figuran a continuación, una copiará la pantalla en la dirección 40000 y otra la tomarán de dicha dirección. Una subrutina en código máquina que realiza esta misma operación se adjunta al programa "HEX LOADER" (carga de rutinas en código máquina).

```
9000  FOR i = 0 TO 6143
9010  POKE 40000 + i, PEEK (16384 + i)
9020  NEXT i
```

```
9050  FOR i = 0 TO 6143
9060  POKE 16384 + i, PEEK (40000 + i)
9070  NEXT i
```

PROGRAMA PG2. GRAFICOS

```
  50 CLEAR 31087
  60 DIM u(10)
  70 FOR i=0 TO 9
  80 LET u(i+1)=31088+168*i
  90 NEXT i
 100 BORDER 1: PAPER 1: INK 7
 110 GO TO 500
 150 GO SUB 8800
 155 BORDER 1: PAPER 1: INK 7
 160 PRINT AT 15,0;"Anadir caracteres";TAB 31;
     "ASalida caracteres en pantalla DVolcado desde mem. a
     caracter  MCopia de caracter a memoria";TAB 31;"CConstruir
     Pant.";TAB 31;"B";"Grabar prog.";TAB 31;"SParar prog.";
     TAB 31;"H"
 170 INPUT z$: LET z$=CHR$ (CODE z$-32*(z$>"£"))
 180 GO TO 200+300*(z$="I")+800*(z$="D")+1300*(z$="M")+1800*
     (z$="C")+2300*(z$="B")+2800*(z$="S")+3300*(z$="H")+3800*
     (z$="A")
 200 INPUT "Pulsa tecla para seguir";z$
 210 GO TO 150
 500 DATA 3,16,1,240,4,0,3,16,1,31,7,0,1,240,4,16,3,0,1,31,12,
     16,3,0,1,255,4,0,3,16,1,255,7,16,1,31,7,16,1,240,7,16,1,
     255,7,0,1,255,4,16
 510 DATA 2,16,3,255,3,16,3,56,1,255,7,56,1,63,4,56,2,16,3,31,5,
     16,3,255,3,0,3,56,1,255,4,0,3,56,1,248,4,56,2,16,3,240,3,16,
     2,0,3,255,3,16,3,0,1,255,4,56
 520 DATA 3,24,2,248,3,0,3,24,2,31,6,0,2,248,3,24,3,0,2,31,11,24,
     3,0,2,255,3,0,3,24,2,255,6,24,2,31,6,24,2,248,6,24,2,255,
     3,24,3,0,2,255,3,24
 590 GO SUB 8500
 600 RESTORE
 610 PRINT TAB 10; FLASH 1;"Inicializando"
 680 PRINT AT 20,5;"ABCDEFGHIJKLMNOPQRSTU";TAB 5;
 690 POKE 23675,112: POKE 23676,121
 700 LET n=0
 710 FOR i=0 TO 255
 720 IF n=0 THEN  READ n,k
 730 LET n=n-1
 740 POKE 31088+i,k
 750 NEXT i
 760 PRINT "                    "
 800 FOR i=31344 TO 32767
 810 POKE i,0
 820 NEXT i
 900 GO TO 150
1000 CLS : PRINT TAB 12;"GRAFICOS"
1010 PRINT ,,TAB 8;"Salida caracteres"
1020 PRINT AT 4,5;"ABCDEFGHIJKLMNOPQRSTU",,TAB 5;
1030 FOR i=31088 TO 32600 STEP 168
1040 POKE 23675,i-256*INT (i/256)
1050 POKE 23676,INT (i/256)
1060 PRINT "                    ";TAB 5;
1070 NEXT i
1080 PRINT
1090 INPUT "Pulsa tecla para seguir";z$
1100 GO SUB 8500
```

```
1110 GO TO 150
1200 GO TO 200
1500 GO SUB 8800
1510 PRINT " Volcado desde memoria a salida caracteres"
1520 INPUT "Borrar salida de graficos (S/N)?";z$
1530 LET z$=CHR$ (CODE z$-32*(z$>"£"))
1540 IF z$<>"S" AND z$<>"N" THEN  BEEP .2,24: GO TO 1520
1550 IF z$="S" THEN  GO SUB 8500
1560 LET im=0
1570 GO SUB 8000
1580 IF i<>0 THEN  GO TO 1700
1600 LET r=4: LET c=12
1610 FOR j=0 TO 7
1620 LET u=PEEK (15616+8*(CODE z$-32)+j)
1630 GO SUB 1900
1640 NEXT j
1650 GO TO 200
1710 LET r=4: LET c=12
1720 FOR j=0 TO 7
1730 LET u=PEEK (u(i)+j+8*(CODE z$-65))
1740 GO SUB 1900
1750 NEXT j
1790 GO TO 200
1900 FOR k=0 TO 7
1910 LET t=2^(7-k)
1920 IF INT (u/t)=1 AND ATTR (r+j,c+k)=15 THEN  PRINT  INK 1;
     PAPER 7; INVERSE 1;AT r+j,c+k;"■"
1930 LET u=u-t*INT (u/t)
1940 NEXT k
1950 RETURN
2000 GO SUB 8800
2010 PRINT TAB 3;"Paso de caracteres a memoria"
2020 LET im=0: LET r=4: LET c=12
2030 GO SUB 8000
2040 PRINT  FLASH 1;"Espera, un poco"
2060 FOR j=0 TO 7
2070 LET u=0
2080 FOR k=0 TO 7
2090 IF ATTR (r+j,c+k)<>15 THEN  LET u=u+2^(7-k)
2100 NEXT k
2110 POKE u(i)+8*(CODE z$-65)+j,u
2120 NEXT j
2200 GO TO 200
2500 CLS : PRINT TAB 8;"Construir PANTALLA"
2510 PRINT ,,"Comandos :"
2520 PRINT ,,"A-U - dibuja A-U  grupo actual"
2530 PRINT ,,"V   - car. standard(salir=SPACE)"
2540 PRINT ,,"W   - graba pantalla en cinta"
2550 PRINT ,,"X   - carga pantalla de cinta"
2560 PRINT ,,"Y   - cambia otro grupo caract."
2570 PRINT ,,"Z   - menu"
2580 PRINT ,,,,"Pulsa tecla para seguir"
2590 PAUSE 0
2600 CLS : LET i=1: POKE 23675,112: POKE 23676,121
2610 PRINT AT 20,5;"ABCDEFGHIJKLMNOPQRSTU";TAB 5;
     "                              "
2620 INPUT z$: LET z$=CHR$ (CODE z$-32*(z$>"£")):
     IF z$<"A" OR z$>"Z" THEN  BEEP .2,24: GO TO 2610
2630 GO TO 2700+3300*(z$<="U")+3400*(z$="V")+3500*
     (z$="W")+3600*(z$="X")+3700*(z$="Y")
2700 GO SUB 8500
2710 GO TO 150
3000 GO SUB 8800
3010 PRINT TAB 10;"Grabar GRAFICOS"
```

```
3020 PRINT ,,"Nombre programa?"
3030 INPUT z$: IF LEN z$>10 THEN  BEEP .2,24: GO TO 3030
3040 SAVE z$ LINE 3100
3050 PRINT ,,"Grabar caracteres"
3060 SAVE z$CODE 31088,1680
3070 PRINT ,,"Fin grabacion"
3080 GO TO 200
3100 LOAD z$CODE 31088,1680
3110 GO TO 200
3500 GO SUB 8800
3510 PRINT "Programa detenido"
3520 STOP
4000 GO SUB 8800
4010 PRINT TAB 6;"Anadir caracteres"
4020 INPUT "Borras salida graficos?(S/N)";z$
4030 LET z$=CHR$ (CODE z$-32*(z$>"£"))
4040 IF z$<>"S" AND z$<>"N" THEN  BEEP .2,24: GO TO 4020
4050 IF z$="S" THEN  GO SUB 8500
4060 INPUT "Coordenadas (1 to 8), 0=fin ";a,b
4070 IF a<0 OR a>8 OR b<0 OR b>8 THEN  BEEP .2,24: GO TO 4060
4080 IF a=0 OR b=0 THEN  GO TO 200
4090 IF ATTR (12-a,11+b)=15 THEN  PRINT  INK 1; PAPER 7;
     INVERSE 1;AT 12-a,11+b;"■"
4100 GO TO 4060
6000 INPUT "Introduce y,x(como  PRINT )";x;" ";y
6010 IF x<0 OR y<0 OR x>19 OR y>31 THEN  BEEP .2,24: GO TO 6000
6020 PRINT AT x,y;CHR$ (CODE z$+79)
6030 GO TO 2610
6100 INPUT "Introduce carac.,y,x ";z$;" ";x;" ";y
6110 IF z$<" " OR z$>"©" OR x<0 OR y<0 OR x>19 OR y>31 THEN
     BEEP .2,24: GO TO 6100
6120 IF z$=" " THEN  GO TO 2610
6130 PRINT AT x,y;z$
6140 GO TO 6100
6200 INPUT "Nombre pantalla ";z$
6210 IF LEN z$>10 THEN  BEEP .2,24: GO TO 6200
6230 SAVE z$SCREEN$
6240 GO TO 2610
6300 INPUT "Nombre pantalla";z$
6310 IF LEN z$>10 THEN  BEEP .2,24: GO TO 6300
6320 LOAD z$SCREEN$
6340 GO TO 2610
6400 INPUT "Elige grupo caracter.(1 a 10)";i
6410 IF i<1 OR i>10 THEN  BEEP .2,24: GO TO 6400
6420 POKE 23675,u(i)-256*INT (u(i)/256)
6430 POKE 23676,INT (u(i)/256)
6440 GO TO 2610
8000 INPUT "Elige grupo de caracter.(1 a 10)";i
8005 LET a$="A": LET b$="U"
8010 IF i<im OR i>10 THEN  BEEP .2,24: GO TO 8000
8020 IF i<>0 THEN  GO TO 8050
8030 INPUT "Introduce caract. (  to©) ";z$
8040 GO TO 8060
8050 INPUT "Introduce caracter (A to U) ";z$
8060 IF i=0 THEN  LET a$=" "
8070 IF i=0 THEN  LET b$="©"
8080 IF z$<a$ OR z$>b$ THEN  BEEP .2,24: GO TO 8020
8090 IF i=0 THEN  RETURN
8100 POKE 23675,u(i)-256*INT (u(i)/256)
8110 POKE 23676,INT (u(i)/256)
8120 RETURN
8500 CLS : PRINT TAB 12;"GRAFICOS"
8510 PRINT AT 2,12;"12345678"
```

```
8520 FOR i=1 TO 8
8530 PRINT AT 3+i,10;9-i;TAB 21;9-i
8540 NEXT i
8550 PRINT AT 13,12;"12345678"
8560 PLOT 91,75
8570 DRAW 73,0
8580 DRAW 0,73
8590 DRAW -73,0
8600 DRAW 0,-73
8700 RETURN
8800 FOR i=14 TO 21
8810 PRINT AT i,0;"
8820 NEXT i
8830 PRINT AT 14,0;
8840 RETURN
```

16. BOCETOS

Escribí este programa inmediatamente después de terminar el programa MUSICA y por ello, ambos presentan grandes afinidades. Ambos tienen un MENU de opciones y almacenan los datos en una cadena numérica. En ambos casos el usuario puede escribir y rectificar los datos, ejecutar el programa y cambiar el dibujo hasta que esté completamente satisfecho con él. El dibujo (o pieza musical) resultante puede entonces ser grabado independientemente en cassette para una ocasión posterior.

Todas las habilidades para dibujar del ZX SPECTRUM pueden ser utilizadas mediante el comando DRAW tal y como se describe en el capítulo 17 del Manual de Programación Basic para el ZX SPECTRUM. La forma básica de este comando es:

DRAW x, y

donde x e y son movimientos relativos que se desean realizar a partir de la posición actual. Una variante del comando es:

DRAW x, y, a

donde x e y significan lo anteriormente dicho y a es el ángulo en RADIANES que gira la línea para realizar el movimiento, para utilizar más fácilmente este comando he convertido todos los ángulos de radianes a GRADOS porque entiendo que la mayor parte de la gente está más familiarizada con esta forma de expresar los ángulos.

Cuando se ejecuta (RUN) el programa pregunta al usuario por los colores que se desean para el BORDER (parte de la pantalla externa a las 20 x 31 posiciones) y el PAPER (parte interna de la pantalla, fondo en el cual se escriben caracteres en colores mediante INK) y el número de movimientos que se esperan introducir. Entonces el programa establecerá una cadena de 2 dimensiones, la cual se utilizará para almacenar la información siguiente:

b (i, 1) movimiento relativo en dirección x
b (i, 2) movimiento relativo en dirección Y
b (i, 3) El ángulo en grados que se desea gire la línea
b (i, 4) -1 para parar, Ø a 7 para determinar el color del dibujo.
b (i, 5) Coordenada absoluta x actual
b (i, 6) Coordenada absoluta y actual.

El usuario controla el programa por medio del MENU que aparece en la figura FK1, el cual se genera en la línea 200. Introduciendo uno de los comandos de la columna de la derecha el programa ejecutará la subrutina adecuada conforme a la línea 230.

```
BOCETOS SPECTRUM

MENU

Escribir (datos)      W
Revisar  (datos)      R
Imprimir (datos)      P
Dibujar               D
Color fondo y borde   B
Cargar de cinta       L
Grabar en cinta       S
Verificar             V
Detener programa      H
```

Figura FK1. El MENU de opciones para BOCETOS en el SPECTRUM

ESCRIBIR **Introducir W**

Esta opción permite al usuario escribir nuevos movimientos o escribir sobre movimientos que él desee borrar. El programa pregunta por el movimiento en el que desea comenzar: normalmente 1.

El dibujo siempre comienza en el ángulo inferior izquierdo de la pantalla, en las coordenadas x = Ø, y = Ø. Los siguientes movimientos comienzan en el punto en el cual el movimiento anterior paró. La máxima distancia que se puede mover en total es de 255 en la dirección x y 175 unidades en la dirección y. El programa comprueba que el dibujo no se salga de estos límites cuando mueve en línea recta, pero el usuario debe tener cuidado y asegurarse de que sus curvas también se encuentran dentro de los límites.

El programa debe conocer, para dibujar, el movimiento de x, el de y y el ángulo a girar en grados medidos en sentido CONTRARIO a las agujas del reloj. Un ángulo negativo hará que se dibuje en el sentido de las agujas del reloj. Una vez conocido el número de orden del movimiento y la terna de valores anteriores se ha de teclear el COLOR de la línea a dibujar (que debe estar comprendido entre 0 y 7), indicando el color que se encuentra en la fila superior del teclado del SPECTRUM. Si se introduce "U" quiere decir que la pluma que dibuja debe "levantarse" (UP) de la pantalla durante ese movimiento, lo que permite al usuario poder cambiar de área en que se dibuja sin dejar rastro.

REVISAR Y DIBUJAR **Introducir R ó P**

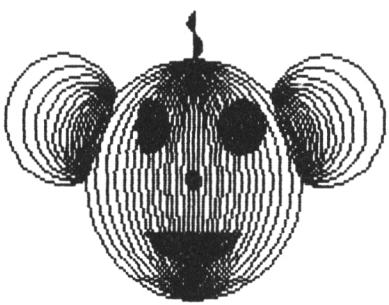
Estas 2 opciones listan las instrucciones actuales en la pantalla o en la impresora. La tabla TK1 lista las instrucciones para dibujar el BOCETO de la FIGURA FK2.

INSTRUCCIONES PARA IMPRESION

Movim.	X	Y	Grados	U/D/C
1	127	40	0	U
2	0	90	180	6
3	0	-90	180	6
4	-45	45	0	U
5	13	32	-270	4
6	77	-32	0	U
7	-13	32	270	4
8	-52	-22	0	U
9	0	20	180	2
10	0	-20	80	2
11	40	0	0	U
12	0	20	180	2
13	0	-20	180	2
14	0	-30	0	U
15	-40	0	-120	4
16	20	16	0	U
17	0	8	150	4
18	0	-8	150	4
19	0	49	0	U
20	0	10	135	4
21	0	8	-130	4

Tabla TK1. Instrucciones físicas utilizadas por el programa "BOCETOS"

FIGURA FK2. Un BOCETO típico



Boceto terminado. Pulsa tecla

COLOR DE FONDO Y BORDE DEL BOCETO — Introducir B

Esta opción permite al usuario cambiar el color de fondo y borde del boceto.

DIBUJO — Introducir D

El programa ejecuta las instrucciones actuales para dibujar el boceto deseado en la pantalla.

PROGRAMA PK1. CROQUIS

```
  1 GO SUB 4999
 10 CLS : PRINT  TAB 8;"BOCETOS SPECTRUM",,,"Introduce:",,
    "  No. de movimientos","  Color del borde (0 - 7)",
    "  Color del fondo (0 - 7)"
 20 INPUT nm,b,p: IF nm<0 OR b<0 OR b>7 OR p<0 OR p>7 THEN
    BEEP .2,0: GO TO 10
 30 BORDER b: PAPER p
 40 LET end=0
 50 DIM b(nm+1,6)
 60 LET b(1,1)=nm
 70 LET b(1,2)=b
 80 LET b(1,3)=p
 90 LET b(1,5)=0
100 LET b(1,6)=0
200 CLS : PRINT TAB 8;"BOCETOS SPECTRUM",,,TAB 14;"MENU",,,TAB 6;
    "Escribir (datos)";TAB 26;"W",TAB 6;"Revisar  (datos)";
    TAB 26;"R",TAB 6;"Imprimir (datos) ";TAB 26;"P",TAB 6;
    "Dibujar";TAB 26;"D",TAB 6;"Color fondo y borde";TAB 26;"B",
    TAB 6;"Cargar de cinta"; TAB 26;"L",TAB 6;"Grabar en cinta";
    TAB 26;"S",TAB 6;"Verificar";TAB 26;"V",TAB 6;
    "Detener programa";TAB 26;"H"
210 INPUT z$
220 IF z$>="a" THEN  LET z$=CHR$ (CODE z$-32)
230 GO TO 300+200*(z$="W")+700*(z$="B")+1200*(z$="R")+1700*
    (z$="L")+2200*(z$="S")+2700*(z$="D")+3200*(z$="V")+3700*
    (z$="H")+4200*(z$="P")
300 PAUSE 200
310 GO TO 200
500 PRINT ,,"          INSTRUCCIONES",
    " Introduce el movimiento a par--tir del cual deseas
    comenzar a  escribir."
510 INPUT j: IF j<1 OR j>nm THEN  GO TO 510
520 PRINT ,," Introduce para cada movimiento las coordenadas
    X e Y, el angulo y U (para subir) o D (para ba--jar).",,
    "Introduce:",,"  F  para terminar","  B  para volver atras"
540 PRINT ,," Movim.  X      Y    Grados U/D/C",,,
545 PRINT ,,
550 FOR i=j+1 TO nm+1
560 PRINT AT 21,3;i-1;TAB 8;"      ": INPUT x$:
    IF x$="f" OR x$="F" THEN  GO TO 900
575 PRINT AT 21,8;x$;"     ";: LET b(i,5)=b(i-1,5)+VAL x$
580 PRINT AT 21,14;"      ";: INPUT y$: LET b(i,6)=b(i-1,6)+VAL y$:
    IF y$="b" OR y$="B" OR b(i,5)>255 OR b(i,5)<0 OR b(i,6)>175
    OR b(i,6)<0 THEN  BEEP .2,0: GO TO 560
585 PRINT AT 21,14;y$;"     ";
590 PRINT AT 21,20;"      ": INPUT z$: IF z$="b" OR z$="B" THEN
    BEEP .2,0: GO TO 580
```

```
595 PRINT AT 21,20;z$;"     ";
600 PRINT AT 21,27;"■";: INPUT a$: IF a$="b" OR a$="B" THEN
    BEEP .2,0: GO TO 590
605 PRINT AT 21,27;a$
610 IF a$<>"u" AND a$<>"d" AND a$<>"U" AND a$<>"D" AND (a$<"0"
    OR a$>"7") THEN  BEEP .2,0: GO TO 600
620 LET b(i,1)=VAL x$: LET b(i,2)=VAL y$: LET b(i,3)=VAL z$:
    LET b(i,4)=-1*(a$="u" OR a$="U")+(CODE a$-48)*(a$<"U")
650 PRINT
660 NEXT i
680 LET end=nm+1
690 GO TO 300
900 IF i>=end THEN  LET end=i-1: GO TO 200
910 FOR j=i TO end
920 LET b(j,5)=b(j-1,5)+b(j,1)
930 LET b(j,6)=b(j-1,6)+b(j,2)
940 NEXT j
950 GO TO 200
1000 PRINT ,,TAB 7;"Cambio de colores",,,"Los valores son:",
     "  Borde",b,"  Fondo",p,,,"Introduce los nuevos valores"
1010 INPUT b,p
1020 BORDER b
1030 PAPER p
1040 GO TO 70
1500 PRINT ,,TAB 6;"Instrucciones",,,
     " Movim.  X     Y   Grados U/D/C"
1505 PRINT ,,
1510 FOR i=2 TO end
1520 PRINT TAB 3;i-1;TAB 8;b(i,1);TAB 14;b(i,2);TAB 22;b(i,3);
     TAB 28;CHR$ (68+17*(b(i,4)=-1)-(20-b(i,4))*(b(i,4)>=0))
1530 NEXT i
1540 PRINT ,,"Pulsa para volver al MENU"
1550 PAUSE 32768
1560 GO TO 200
2000 PRINT ,,TAB 10;"Extraer un boceto";AT 21,0;
     "Introduce el nombre del boceto"
2010 INPUT z$: IF LEN z$>10 THEN  PRINT AT 21,0;
     "Nombre muy largo. Repitelo": GO TO 2010
2015 PRINT AT 21,0;"                          "
2020 PRINT AT 21,0; FLASH 1;"CARGANDO PROGRAMA"
2030 LOAD z$ DATA b()
2040 LET nm=b(1,1)
2050 LET b=b(1,2)
2060 LET p=b(1,3)
2070 LET end=b(1,4)
2080 BORDER b
2090 PAPER p
2100 PRINT AT 19,0;z$;"   Carga correcta              "
2110 GO TO 300
2500 PRINT ,,TAB 9;"Grabar un boceto";AT 21,0;
     "Introduce el nombre"
2510 INPUT z$: IF LEN z$>10 THEN  PRINT AT 21,0;
     "Nombre muy largo. Repitelo": GO TO 2510
2520 LET b(1,4)=end
2530 PRINT AT 21,0; FLASH 1;"REALIZANDO GRABACION"
2540 SAVE z$ DATA b()
2550 PRINT AT 21,0;z$;"  Grabacion correcta              "
2560 GO TO 300
3000 CLS
3010 PAUSE 100
3020 FOR i=2 TO end
3021 IF b(i,4)=-1 THEN  GO TO 3035
3022 FOR j=0 TO b(i,3) STEP b(i,3)/15
```

```
3024 IF i=2 THEN  PLOT 0,0: GO TO 3030
3026 PLOT  INK b(i,4);b(i-1,5),b(i-1,6)
3030 DRAW  INK b(i,4);b(i,1),b(i,2),j*PI/180
3032 NEXT j
3033 GO TO 3040
3035 PLOT b(i,5),b(i,6)
3040 NEXT i
3050 PRINT AT 21,0; FLASH 1;"Boceto terminado. Pulsa  tecla"
3060 PAUSE 0
3070 GO TO 200
3500 PRINT ,,TAB 8;"Verificar un boceto";AT 21,0;
     "Introduce el nombre del boceto"
3510 INPUT z$: IF LEN z$>10 THEN  BEEP .2,0: PRINT AT 21,0;
     "Nombre muy largo. Repitelo": GO TO 3510
3520 PRINT AT 21,0; FLASH 1;"  REALIZANDO   VERIFICACION     "
3530 VERIFY z$ DATA b()
3540 PRINT AT 19,0;z$;"  Verificacion   correcta       "
3550 GO TO 300
4000 PRINT ,,"Programa detenido"
4010 STOP
4500 PRINT AT 21,0; FLASH 1;"REALIZANDO IMPRESION":
     LPRINT ,,TAB 7;"Imprimir instrucciones":
     LPRINT ,," Movim.  X     Y   Grados U/D/C"
4505 PRINT ,,
4510 FOR i=2 TO end
4520 LPRINT TAB 3;i-1;TAB 8;b(i,1);TAB 14;b(i,2);TAB 22;b(i,3);
     TAB 28;CHR$ (68+17*(b(i,4)=-1)-(20-b(i,4))*(b(i,4)>=0))
4530 NEXT i
4535 PAUSE 0
4540 GO TO 200
4999 CLS : PRINT AT 8,6;"BOCETOS SPECTRUM"
5000 : PRINT AT 10,0;"Este programa permite represen- tar dibujos
     o bocetos en panta- lla y en alta resolucion.","  Ello se
     conseguira a traves   de rectas y curvas introducien- do el
     punto donde comienzan asi","como el angulo de inclinacion. ",
     "  En el apartado U/D/C se in-  ","troducira U, D o un color
     (1-6)"
5010 PAUSE 0
5020 RETURN
```

17. ESPIRALES

Espirales es un programa para potenciar la habilidad, talento artístico y el ingenio del propietario del SPECTRUM. No hay límite alguno para la variedad de curvas y espirales que puede dibujar el programa, salvo las impuestas por la imaginación del usuario. El programa proporciona al usuario las instrucciones necesarias para seleccionar y dibujar cada una de las 8 curvas ejemplo que se incluyen, para experimentar con una curva por sí mismo y, cuando esté satisfecho de lo realizado almacenando y grabando un cassette para una ocasión posterior.

ESTRUCTURA DEL PROGRAMA

Cuando se ejecuta (RUN), el programa crea la cadena literal f$, en la cual las ecuaciones a dibujar se almacenarán de forma permanente. A continuación lee los 8 pares de ecuaciones (que se toman como ejemplo) de la cadena f$, los cuales se encuentran en las sentencias DATA de las líneas 200 a 270, enviándolas a los almacenamientos temporales x$ e y$.

El programa debe almacenar también la longitud (en caracteres) de cada ecuación, de forma que pueda tomarlas correctamente de la cadena f$. Un método mediante el cual podría realizarse esto sería crear una cadena numérica e introducir la longitud de cada ecuación dentro de los elementos de esa cadena. Este método tiene la desventaja de que podría haber dos limitaciones en el números de ecuaciones que es capaz de almacenar el programa: la primera limitación podría ser la longitud de f$ y la segunda podría ser el número de elementos en la cadena. Para evitar este problema el programa reserva dentro de f$ el byte precedente a cada ecuación para la longitud de la ecuación en caracteres. De esta forma la longitud de f$ pasa a ser la única limitación en la capacidad del programa y ella puede ser aumentada mediante el incremento del valor de (tl) en la línea 110. La única desventaja es que la longitud de la ecuación es almacenada en un solo byte y por ello el número máximo de ecuaciones posibles es de 255.

Teniendo almacenadas las ecuaciones del ejemplo en f$ el programa imprime el MENU ilustrado en la Figura FP1. El usuario puede elegir ver una de las curvas del ejemplo seleccionando la opción "DIBUJO DE UNA CURVA". El programa imprime las ecuaciones actualmente almacenadas en f$ y el usuario elige una de ellas. El programa se transfiere a las ecuaciones x$ e y$, va a la subrutina de la línea 1600 y dibuja la curva seleccionada, continúa haciendo esto hasta que el usuario presione alguna tecla, en cuyo caso vuelve al MENU de opciones.

```
                    ESPIRALES

                       MENU

Introducir una funcion          T
Dibujar    una funcion          D
Almacenar  una funcion          M
Grabar y verif. el programa     S
Detener el programa             H
```

FIGURA FP1. **El MENU de opciones de ESPIRALES**

El usuario puede introducir un nuevo par de ecuaciones seleccionando la opción "INTRODUCIR UNA ECUACION". El computador no comprueba la sintáxis de las ecuaciones y por tanto se debe tener cuidado y asegurarse de que no hay caracteres ilegales y que los paréntesis están dispuestos simétricamente, etc. si se comete algún error el programa se detendrá e imprimirá el código de error "c sin sentido en BASIC 1740:1". La situación puede ser subsanada haciendo GOTO 1000 para obtener el menú de opciones, seleccionando la opción "introducir una ecuación" una segunda vez (después de corregida).

Las ecuaciones introducidas por el usuario son almacenadas temporalmente en x$ e y$, de tal forma que si se introduce un segundo par de ecuaciones el par primero se pierde. Para evitar esto y retener en la memoria el primer par de ecuaciones el usuario debe seleccionar la opción "ALMACENAR UNA FUNCION". Esto hace que el programa transfiera el contenido actual de x$ e y$ a f$, de tal forma que la nueva curva sea permanentemente accesible mediante la opción "DIBUJAR UNA FUNCION". De esta manera el usuario puede almacenar una colección de curvas a base de introducir ecuaciones (opción "T"), cambiándolas hasta que esté satisfecho y almacenarlas después.

Hemos de llamar la atención sobre el punto de que, si un programa se detiene ("H") y se vuelve a ejecutar (RUN), se borran todas las variables y vuelven a generarse de nuevo. Así, las ecuaciones que el usuario almacena no se encuentran siempre de la misma forma como están en las 8 ecuaciones del ejemplo, puesto que estas ecuaciones se vuelven a cargar a partir de las sentencias DATA cuando el programa se ejecuta (RUN). De aquí que se incluya como opción "grabar en cinta y verificar el programa", que graba el programa en cassette, incluidas todas sus variables. Cuando el programa grabado es vuelto a cargar comienza automáticamente en la línea 1000, se imprime el MENU de opciones con todas las variables y todas las ecuaciones de f$ todavía intactas.

LAS CURVAS EJEMPLO

Es sorprendentemente difícil predecir el tamaño de una curva a partir de sus ecuaciones PARAMETRICAS. El círculo es la ecuación fundamental para todas las curvas cerradas, elipses y espirales y las ecuaciones de un círculo de radio 1 son:

$$x = \cos n$$
$$y = \sin n$$

Cuando la dibujamos utilizando el programa ESPIRALES, aparece un pequeño círculo en el cuadro de la pantalla con centro en x = 127 e y = 70. De igual manera, un círculo de radio 60 tiene de ecuaciones:

$$x = 60 * \cos n$$
$$y = 60 * \sin n$$

Los valores máximos y mínimos de las funciones seno (SIN) y coseno (COS) son -1 y +1 y así, el círculo más grande con centro en x = 127 e y = 70 que puede ser dibujado sin salirse de la pantalla es:

$$x = 70 * \cos n$$
$$y = 70 * \sin n$$

Si se necesita dibujar un círculo con un radio mayor que 87 debemos mover el centro del círculo en la dirección del eje y hasta y = 87 mediante una pequeña modificación de la línea 1730 del programa. El círculo podía entonces invadir la parte de la pantalla utilizada para escribir el título y las ecuaciones.

Una elipse es como un círculo que se ha alargado en una de las 2 direcciones. La elipse mayor que puede dibujarse en una pantalla con centro en x = 127, y = 70 tiene de ecuaciones:

$$x = 127 * \cos n$$
$$y = 70 * \sin n$$

Para girar una elipse un ángulo determinado es necesario "mezclar" las funciones seno y coseno en las ecuaciones paramétricas, como por ejemplo:

$$x = 30 * \cos n + 15 * \sin n$$
$$y = 30 * \sin n + 15 * \cos n$$

Los efectos interesantes se obtienen mediante la mezcla de dos círculos simultáneamente como aparecen en la segunda curva, ilustrada en la figura FP2 las ecuaciones son:

$$x = 25 * \cos (12 * n) + 50 * \cos n$$
$$y = 15 * \sin (12 * n) + 30 * \sin n$$

ESPIRALES

```
25*COS (12*n)+50*COS n
15*SIN (12*n)+30*SIN n
```

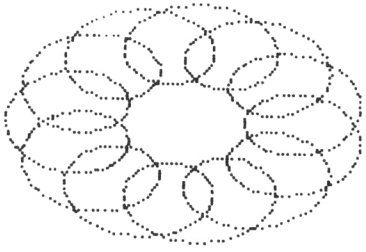


FIGURA FP2. **Una gran elipse con 12 pequeñas elipses superpuestas (ver texto)**

La forma básica de la figura es una elipse de 50 unidades de largo y 30 unidades de ancho tal y como se deduce del segundo término de cada una de las ecuaciones x e y. Superpuestas a esta figura se dibujan 12 pequeñas elipses de 25 unidades de longitud y 15 de alto, tal y como se deduce del primer término de las ecuaciones paramétricas. En las Figuras FP3 y FP4 aparecen otras curvas.

Una guía para conocer el tamaño de una figura puede obtenerse usando los coeficientes multiplicadores de las funciones seno (SIN) y coseno (COS). Se ha de tener cuidado de que la suma de dichos coeficientes ser menor de 127 en la dirección x y de 70 en la dirección y pues de otra forma el dibujo rebasará los límites de la pantalla.

Se pueden usar otras funciones en lugar de éstas o funciones que incorporen una combinación de las funciones seno y coseno, pero se ha de tener cuidado de lo salirse de la pantalla y de que las curvas sean CERRADAS (aquellas que una vez y otra recorren el mismo camino), es decir: deben tener un límite superior y otro inferior y deben arrojar los mismos valores de x e y para valores homólogos de n en distintos ciclos.

La función seno (SIN) por ejemplo nunca sobrepasa el valor +1 y nunca es inferior a -1, siendo sin (n + 2 π) = sin n, etc.

```
ESPIRALES

70*COS (n/8)*COS n
50*SIN (n/8)*SIN n
```

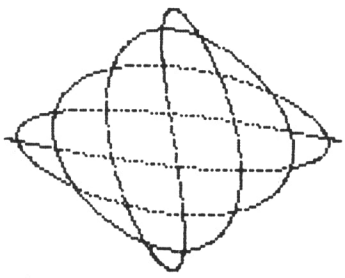


FIGURA FP3. **Una figura de las funciones cos y sen simultáneamente**

ESPIRALES

15*COS (10*n)+80*COS n+5*COS (n/4)
15*SIN (10*n)+45*SIN n+5*SIN (n/4)

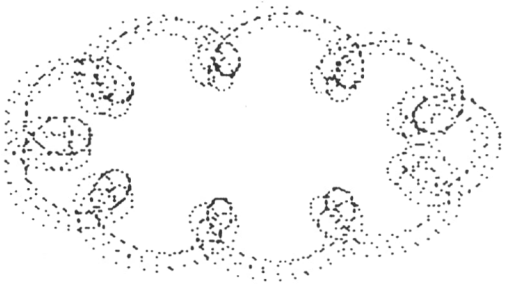


FIGURA FP4. Una gran elipse con elipses más pequeñas superpuestas y perturbada por un pequeño término (cos) y (sin)

PROGRAMA PP1. ESPIRALES

```
  5 CLS : PRINT TAB 11;"ESPIRALES"
 10 GO SUB 4000
100 BORDER 1: PAPER 1: INK 7
110 LET olf=0: LET lf=0: LET tl=1000
120 DIM f$(tl)
200 DATA 8
210 DATA "50*COS n+50*COS (n/4)","30*SIN n+30*SIN (n/4)",
    "25*COS (12*n)+50*COS n","15*SIN (12*n)+30*SIN n"
220 DATA "50*COS n+50*SIN (n/6)","30*SIN n+30*COS (n/6)"
230 DATA "20*COS n+20*COS (n/2)+20*COS (n/4)","15*SIN n+15*
    SIN (n/2)+15*SIN (n/4)"
240 DATA "70*COS (n/8)*COS n","50*SIN (n/8)*SIN n"
250 DATA "50*COS (n/8)*COS n+50*COS (n/4)","35*SIN (n/8)*
    SIN n+35*SIN (n/4)"
260 DATA "40*SIN n+20*SIN (4*n)+10*SIN (16*n)+5*SIN (32*n)",
    "32*COS n+8*COS (4*n)+4*COS (16*n)+2*COS (32*n)"
270 DATA "15*COS (10*n)+80*COS n+5*COS (n/4)","15*SIN (10*n)+45*
    SIN n+5*SIN (n/4)"
300 READ nf
310 FOR i=1 TO nf
320 READ x$,y$
330 IF LEN x$>255 OR LEN y$>255 THEN  BEEP .2,24:
    PRINT "La funcion en la instruccion DATA tiene mas de
    255 caracteres": STOP
340 LET lf=lf+LEN x$+LEN y$+2
350 IF lf>=1000 THEN  BEEP .2,24:
    PRINT ,,"No hay espacio suficiente para la funcion -
    incrementa tl en la linea 110": STOP
360 LET f$(olf+1 TO lf)=CHR$ LEN x$+x$+CHR$ LEN y$+y$
370 LET olf=lf
380 NEXT i
390 LET x$="": LET y$=""
1000 LET t=28: CLS : PRINT TAB 11;"ESPIRALES"
1010 PRINT ,,,,TAB 14;"MENU"
```

```
1020 PRINT ,,"Introducir una funcion";TAB t;"T",
     "Dibujar    una funcion";TAB t;"D","Almacenar  una funcion";
     TAB t;"M","Grabar y verif. el programa";TAB t;"S",
     "Detener el programa";TAB t;"H"
1030 INPUT z$: LET z$=CHR$ (CODE z$-32*(z$>"£"))
1040 GO TO 1200+300*(z$="T")+800*(z$="D")+1300*(z$="M")+1800*
     (z$="S")+2300*(z$="H")
1200 INPUT "Pulsa una tecla para continuar";z$
1210 GO TO 1000
1500 PRINT ,,TAB 9;"Introduce una funcion"
1510 INPUT "Introduce la ecuacion en funcion de n para el eje OX";
     x$
1520 IF LEN x$>255 THEN  BEEP .2,24:
     PRINT ,,"La ecucion debe tener menos de 255 caracteres.
     Introduce otra.": GO TO 1510
1530 PRINT ,,,,"La ecuacion es :",,x$
1550 INPUT "Introduce la ecuacion en funcion de n para el eje OY";
     y$
1560 IF LEN y$>255 THEN  BEEP .2,24:
     PRINT ,,"La ecuacion debe tener menos de 255 caracteres.
     Introduce otra.": GO TO 1550
1570 PRINT ,,,,"La ecuacion es :",,y$
1580 GO SUB 1600
1590 GO TO 1200
1600 INPUT "Quieres una copia en la impresora? (S/N)";z$
1610 LET z$=CHR$ (CODE z$-32*(z$>"£"))
1700 CLS : PRINT TAB 11;"ESPIRALES"
1710 PRINT ,,x$: PRINT y$
1730 LET n=0: LET cx=127: LET cy=70
1740 LET sx=cx+VAL x$: LET sy=cy+VAL y$
1750 LET rx=VAL x$: LET ry=VAL y$
1760 PLOT cx+rx,cy+ry
1770 LET n=n+.025
1780 IF INKEY$<>"" THEN  GO TO 1800
1790 GO TO 1750
1800 IF z$="S" THEN  COPY
1850 RETURN
2000 PRINT ,,"Dibujar una funcion"
2010 PRINT ,,"Las funciones validas son :"
2020 LET f=0: LET n=1
2030 GO SUB 2400
2040 IF s>=lf THEN  GO TO 2100
2050 PRINT ,,n;TAB 6;f$(s TO f);TAB 6;
2060 GO SUB 2400
2070 PRINT f$(s TO f)
2080 LET n=n+1
2090 GO TO 2030
2100 INPUT "Introduce no. de funcion ";m
2110 IF m>n OR n<0 THEN  BEEP .2,24: GO TO 2100
2120 LET f=0: LET n=1
2130 GO SUB 2400
2140 IF n=m THEN  LET x$=f$(s TO f)
2150 GO SUB 2400
2160 IF n<>m THEN  LET n=n+1: GO TO 2130
2170 LET y$=f$(s TO f)
2180 GO SUB 1600
2190 LET x$=""
2200 LET y$=""
2210 GO TO 1200
2400 LET s=f+2
```

```
2410 LET f=s+CODE f$(s-1)-1
2420 RETURN
2500 PRINT ,,;TAB 8;"Almacenar una funcion"
2510 IF LEN x$=0 OR LEN y$=0 THEN  BEEP .2,24:
     PRINT ,,"La funcion no esta definida": GO TO 1200
2520 LET lf=lf+LEN x$+LEN y$+2
2530 IF lf>=1000 THEN  BEEP .2,24:
     PRINT ,"No hay suficiente espacio para almacenar la funcion":
     LET lf=lf-LEN x$-LEN y$-2: GO TO 1200
2540 LET f$(olf+1 TO lf)=CHR$ LEN x$+x$+CHR$ LEN y$+y$
2550 LET olf=lf
2560 PRINT ,,"Funcion almacenada"
2570 GO TO 1200
3000 PRINT ,,TAB 8;"Grabar ESPIRALES"
3010 SAVE "ESPIRALES" LINE 1000
3020 INPUT "Deseas verificar lo grabado? (S/N)";z$
3030 IF z$<>"S" AND z$<>"s" THEN  GO TO 1200
3040 PRINT ,,"Rebobina la cinta",,,,
     "                 Inserta la clavija ""EAR""",,,
     "Pulsa PLAY en el cassette"
3050 LET z$="S"
3060 VERIFY "ESPIRALES"
3070 PRINT ,,"Copia verificada"
3080 GO TO 1200
3500 PRINT ,,"Programa terminado"
3510 STOP
4000 PRINT ,,,,,,,,,,
     "  Este programa permite representar en pantalla funciones,
     que  se introduciran independientemente para las coordenadas
     X e Y.   Ambas deberan expresarse en funcion de n
     (parametricas)."
4010 PRINT "                                    Tambien podras
     dibujar una de las 8 graficas ya definidas, grabar el
     programa o sacar una co- pia en la impresora.",,
4050 PRINT ,,, FLASH 1;"PULSA UNA TECLA PARA COMENZAR"
4060 IF INKEY$="" THEN  GO TO 4060
4070 RETURN
```

18. LA CAZA DEL PATO

La compleja estructura de la salida del fichero de pantalla en el SPECTRUM, tal y como se describe en el capítulo 24 del Manual de Programación Basic para el ZX SPECTRUM, hace más bien difícil manipular el display a una velocidad satisfactoria utilizando un programa en Basic.

Este programa desarrolla una técnica para modificar la salida por pantalla a una razonable velocidad por medio de cambios en las características del fichero. Una segunda versión del programa utiliza una corta subrutina en código máquina para acelerar el movimiento.

El programa simula un puesto para la caza del pato similar a la realidad. Cuatro líneas de patos desfilan por la pantalla de arriba a abajo y de izquierda a derecha (y viceversa) hasta que desaparecen.

El jugador tiene una sola escopeta situada en la parte inferior de la pantalla, pudiendo moverla mediante las teclas 5 y 8 y dispararla mediante la tecla número 7. Su objetivo es acertar a todos los patos de la pantalla utilizando el menor número posible de disparos y movimientos de la escopeta, todo ello en el menor tiempo posible.

El programa PU1 está todo él escrito en BASIC. La primera línea llama a la subrutina 7000, la cual establece 4 caracteres gráficos definidos por el usuario y que representan la bala (B), la escopeta (G) y un pato moviéndose a la derecha (D) y a la izquierda (E). El jugador elige a continuación los colores de la pantalla (PAPER) y de los caracteres que en ella se describen (INK), p e i respectivamente y el programa produce un "AREA DE JUEGO" tal y como se ilustra en la figura FU1.

FIGURA FU1. **La salida por pantalla de "LA CAZA DEL PATO"**

Hemos de hacer notar que los comandos que dibujan (PRINT) los patos (líneas 200 a 290) no son muy normales: cada una de las 8 líneas dibujadas está compuesta por una secuencia repetida de 1 pato en el color i (definido mediante INK) y de 3 patos en el mismo color p que se ha definido para la pantalla. Estos 3 últimos patos se vuelven así INVISIBLES y en la salida por pantalla sólo se nota una cuarta parte de los patos que se encuentran realmente allí.

El juego en sí se controla mediante el bucle situado entre las líneas 2000 y 2110. Este bucle llama a la subrutina de la línea 2200, la cual tiene como misión tomar de forma rotativa cada uno de los 32 bytes de una fila de pantalla para que los patos vayan a la derecha o a la izquierda. El valor de la rotación (d) influye en el color i (que define INK) y el efecto que produce es crear la impresión de movimiento de los patos al colorear rotativa e independientemente con p uno de cada 4 patos en pantalla y cada una de las 4 líneas.

La subrutina de la línea 2200 llama asimismo a otras subrutinas.

La función de la primera, en la línea 3000, es borrar la bala de la pantalla previamente al movimiento de los patos. La segunda subrutina, en la línea 3100, restituye la bala después de que se han movido los patos y la hace moverse hacia arriba. Si un determinado pato ocupa la posición del carácter en el cual se encuentra la bala tanto el pato como la bala se borran mediante el campo del color del pato de i a p. La subrutina de la línea 4000 hace disparar la escopeta (*).

La segunda versión del programa (PU2) utiliza una subrutina en código máquina para realizar la rotación que produce la impresión de movimiento. Esta subrutina se almacena desde el comando DATA de la línea 20 en una parte de memoria situada entre el RAMTOP y el área reservada para los gráficos definidos por el usuario, lo cual se realiza mediante el bucle de las líneas 30 a 60. La rutina en código máquina se lista en la tabla TU1.

HEXADECIMAL		**MNEMONICOS** (ENSAMBLADOR)
06 02		LD B, 2
21 00 58		LD HL, 22528
7E	NEXT	LD A, (HL)
C5		PUSH BC
54		LD D, H
5D		LD E, L
23		INC HL
01 1F 00		LD BC, 31
ED B0		LD IR
2B		DEC HL
77		LD (HL), A
01 40 00		LD BC, 64
09		ADD AL, BC
7E		LD A, (HL)
54		LD D, H
5D		LD E, L
2B		DEC HL
01 1F 00		LD BC, 31
ED B8		LD DR
23		INC HL
77		LD (HL), A
01 40 00		LD BC, 64
09		ADD HL, BC
C1		POP BC
10 DE		DJNZ, —34 NEXT
C9		RET

TABLA TU1. **Esta subrutina cambia las líneas 0, 2, 4, 6 del archivo de características**

(*) NOTA DEL TRADUCTOR

La puntuación en cada momento se obtiene mediante la línea 2280.
Si no se es muy rápido con los cursores la puntuación puede ser negativa, lo cual puede evitarse cambiando el 50 de la línea 2270 (tiempo transcurrido) por un número menor.

PROGRAMA PU1. **Versión BASIC de la "CAZA DEL PATO"**

```
  10 GO SUB 7000
 100 PRINT "Color de fondo ?": INPUT p:
     IF p<0 OR 0>7 THEN  BEEP 1,12: GO TO 100
 110 PRINT "Color de los patos ?": INPUT i:
     IF i<0 OR i>7 OR i=p THEN  BEEP 1,12: GO TO 110
 120 BORDER p: PAPER p: INK i: CLS
 130 LET real=8*p+i
 140 LET l=20: LET cg=16: LET h=0: LET cy=0
 150 PRINT AT l,cg;"G"
 160 PRINT AT 0,0;
 200 FOR j=1 TO 2
 210 FOR k=1 TO 8
 220 PRINT  INK i;"E"; INK p;"EEE";
 230 NEXT k
 240 PRINT  INK p,,
 250 FOR k=1 TO 8
 260 PRINT  INK p;"D"; INK i;"D"; INK p;"DD";
 270 NEXT k
 280 PRINT  INK p,,
 290 NEXT j
 300 PRINT  INK i,,,,,,,,,,,,,,,,,,,,,,,,,,
2000 LET s=22528
2010 FOR j=1 TO 2
2020 LET d=1
2030 GO SUB 2200
2040 IF h>=32 THEN  PRINT AT 5,9; FLASH 1;"Fin del juego":
     FOR z=1 TO 500: NEXT z: RUN
2050 LET s=s+95
2060 LET d=-1
2070 GO SUB 2200
2080 IF h>=32 THEN  PRINT AT 5,9; FLASH 1;"Fin del juego":
     FOR z=0 TO 500: NEXT z: RUN
2090 LET s=s+33
2100 NEXT j
2110 GO TO 2000
2200 LET a=PEEK s
2210 FOR i=0 TO 30*d STEP d
2215 GO SUB 3000
2220 POKE s+i,PEEK (s+i+d)
2230 GO SUB 3100
2240 GO SUB 4000
2250 NEXT i
2260 POKE s+d*31,a
2270 LET cy=cy+50
2280 PRINT AT 21,0;"Aciertos = ";h,"Puntos = ";h*100-cy,
2290 RETURN
3000 IF l=20 THEN  RETURN
3010 PRINT  INK p; PAPER p;AT l,c;" "
3020 IF l=0 OR l=4 THEN  PRINT AT l,c; INK p; PAPER p;"E"
3030 IF l=2 OR l=6 THEN  PRINT AT l,c; INK p; PAPER p;"D"
3040 LET l=l-2: RETURN
3100 IF l<0 THEN  LET l=20: RETURN
3110 IF l=20 THEN  RETURN
3120 IF l>7 THEN  PRINT AT l,c;"B": RETURN
3130 IF ATTR (l,c)=real THEN  PRINT AT l,c; INK p; PAPER p;" ":
     LET h=h+1: LET l=20: RETURN
3140 PRINT AT l,c;"B": RETURN
3150 RETURN
4000 LET k=PEEK 23560: POKE 23560,0
```

```
4010 GO TO 4100+100*(k=55)+200*(k=53)+300*(k=56)
4100 RETURN
4200 IF l<>20 THEN  RETURN
4210 LET l=l-2: LET c=cg: LET cy=cy+30
4220 PRINT AT l,c;"B"
4230 RETURN
4300 IF cg<>0 THEN  LET cg=cg-1
4310 PRINT AT 20,cg+1;" "
4320 PRINT AT 20,cg;"G"
4330 LET cy=cy+5
4340 RETURN
4400 IF cg<>31 THEN  LET cg=cg+1
4410 PRINT AT 20,cg-1;" "
4420 PRINT AT 20,cg;"G"
4430 LET cy=cy+5
4440 RETURN
7000 POKE USR "B",0
7001 POKE USR "B"+1,0
7002 POKE USR "B"+2,16
7003 POKE USR "B"+3,16
7004 POKE USR "B"+4,16
7005 POKE USR "B"+5,16
7006 POKE USR "B"+6,0
7007 POKE USR "B"+7,0
7008 POKE USR "G",0
7009 POKE USR "G"+1,24
7010 POKE USR "G"+2,24
7011 POKE USR "G"+3,24
7012 POKE USR "G"+4,24
7013 POKE USR "G"+5,24
7014 POKE USR "G"+6,24
7015 POKE USR "G"+7,0
7016 POKE USR "D",0
7017 POKE USR "D"+1,12
7018 POKE USR "D"+2,14
7019 POKE USR "D"+3,4
7020 POKE USR "D"+4,60
7021 POKE USR "D"+5,124
7022 POKE USR "D"+6,60
7023 POKE USR "D"+6,0
7024 POKE USR "E",0
7025 POKE USR "E"+1,48
7026 POKE USR "E"+2,112
7027 POKE USR "E"+3,32
7028 POKE USR "E"+4,60
7029 POKE USR "E"+5,62
7030 POKE USR "E"+6,60
7031 POKE USR "E"+7,0
7032 RETURN
```

PROGRAMA PU2. Esta versión de la "CAZA DEL PATO" utiliza la rutina en CODIGO MAQUINA para manejar el FICHERO de atributos

```
  10 CLEAR 32559
  20 DATA 6,2,33,0,88,126,197,84,93,35,1,31,0,237,176,43,119,
     1,64,0,9,128,84,93,43,1,31,0,237,184,35,119,1,64,0,9,193,
     16,222,201
  30 FOR i=0 TO 39
  40 READ j
  50 POKE 32560+i,j
  60 NEXT i
  70 GO SUB 7000
  80 LET r=0
  90 PRINT "LA CAZA DEL PATO"
  95 PRINT
 100 PRINT "Introduce color de fondo (0-7)": INPUT p:
     IF p<0 OR p>7 THEN  BEEP 1,12: GO TO 100
 105 PRINT
 110 PRINT "Introduce color de patos (0-7)": INPUT i:
     IF i<0 OR i>7 OR i=p THEN  BEEP 1,12: GO TO 110
 120 BORDER p: PAPER p: INK i: CLS
 130 LET real=8*p+i
 140 LET l=20: LET cg=18: LET h=0: LET cy=0
 150 PRINT AT l,cg;"▌"
 160 PRINT AT 0,0;
 200 FOR j=1 TO 2
 210 FOR k=1 TO 8
 220 PRINT  INK i;"▙"; INK p;"▙▙▙";
 230 NEXT k
 240 PRINT  INK p,,
 250 FOR k=1 TO 8
 260 PRINT  INK p;"▟"; INK i;"▟"; INK p;"▟▟";
 270 NEXT k
 280 PRINT  INK p,,
 290 NEXT j
 300 PRINT  INK i,,,,,,,,,,,,,,,,,,,,,,,,,,
2000 LET s=22528
2010 FOR j=1 TO 2
2020 LET d=1
2030 GO SUB 2200
2040 IF h>=32 THEN  CLS : PRINT AT 5,5; FLASH 1;"Se acabo tu vida.
     Pulsa la tecla 1": IF INKEY$="1" THEN  RUN
2050 LET s=s+95
2060 LET d=-1
2070 GO SUB 2200
2080 IF h>=32 THEN  CLS : PRINT AT 5,5; FLASH 1;"Se acabo tu vida.
     Pulsa la tecla 1": IF INKEY$="1" THEN  RUN
2090 LET s=s+33
2100 NEXT j
2110 GO TO 2000
2200 LET a=PEEK s
2210 FOR i=0 TO 30*d STEP d
2215 GO SUB 3000
2220 POKE s+i,PEEK (s+i+d)
2230 GO SUB 3100
2240 GO SUB 4000
2250 NEXT i
2260 POKE s+d+31,a
2270 LET cy=cy+50
2280 PRINT AT 21,0;"Impactos = ";h,"Puntos = ";h*100-cy,
2290 RETURN
3000 IF l=20 THEN  RETURN
3010 PRINT  INK p; PAPER p;AT l,c;" "
3020 IF l=0 OR l=4 THEN  PRINT AT l,c; INK p; PAPER p;"▙"
3030 IF l=2 OR l=6 THEN  PRINT AT l,c; INK p; PAPER p;"▟"
```

```
3040 LET l=l-2: RETURN
3100 IF l<0 THEN  LET l=20: RETURN
3110 IF l=20 THEN  RETURN
3120 IF l>7 THEN  PRINT AT l,c;"▝": RETURN
3130 IF (l=0 OR l=4) AND ATTR (l,c)=real THEN  PRINT AT l,c;
     INK p;; PAPER p;" ": LET h=h+1: LET l=20: RETURN
3135 IF (l=2 OR l=6) AND ATTR (l,c)=real THEN  PRINT AT l,c;
     INK p;; PAPER p;" ": LET h=h+1: LET l=20: RETURN
3140 PRINT AT l,c;"▝": RETURN
3150 RETURN
4000 LET k=PEEK 23560: POKE 23560,0
4010 GO TO 4100+100*(k=55)+200*(k=53)+300*(k=56)
4100 RETURN
4200 IF l<>20 THEN  RETURN
4210 LET l=l-2: LET c=cg: LET cy=cy+30
4220 PRINT AT l,c;"▗"
4230 RETURN
4300 IF cg<>0 THEN  LET cg=cg-1
4310 PRINT AT 20,cg+1;" "
4320 PRINT AT 20,cg;"▌"
4330 LET cy=cy+5
4340 RETURN
4400 IF cg<>31 THEN  LET cg=cg+1
4410 PRINT AT 20,cg-1;" "
4420 PRINT AT 20,cg;"▌"
4430 LET cy=cy+5
4440 RETURN
7000 POKE USR "B",0
7001 POKE USR "B"+1,0
7002 POKE USR "B"+2,16
7003 POKE USR "B"+3,16
7004 POKE USR "B"+4,16
7005 POKE USR "B"+5,16
7006 POKE USR "B"+6,0
7007 POKE USR "B"+7,0
7008 POKE USR "G",0
7009 POKE USR "G"+1,24
7010 POKE USR "G"+2,24
7011 POKE USR "G"+3,24
7012 POKE USR "G"+4,24
7013 POKE USR "G"+5,24
7014 POKE USR "G"+6,24
7015 POKE USR "G"+7,0
7016 POKE USR "D",0
7017 POKE USR "D"+1,12
7018 POKE USR "D"+2,14
7019 POKE USR "D"+3,4
7020 POKE USR "D"+4,60
7021 POKE USR "D"+5,124
7022 POKE USR "D"+6,60
7023 POKE USR "D"+6,0
7024 POKE USR "E",0
7025 POKE USR "E"+1,48
7026 POKE USR "E"+2,112
7027 POKE USR "E"+3,32
7028 POKE USR "E"+4,60
7029 POKE USR "E"+5,62
7030 POKE USR "E"+6,60
7031 POKE USR "E"+7,0
7032 RETURN
```

Nota: Puedes tambien sustituir la instruccion
2225 por esta otra sentencia:
LET r=r+1-8*(r=7):IF r=0 THEN RANDOMIZE USR 32560
:LET cy=cy+10
y anadir una linea 80 LET r=0

19. DIBUJO (DIGITISER)

Este programa puede ser utilizado para reproducir en la pantalla el DIBUJO de un mapa, diagrama, poster o pintura. Tiene la particularidad de: reproducir a escala los valores introducidos por el usuario, trazar una línea en pantalla y colorear el área comprendida entre dos partes de dicha línea. Este DIBUJO puede ser modificado por el usuario PIXEL A PIXEL, y las características de la situación de cada PRINT AT pueden ser modificadas directamente desde el programa. Asimismo, el DIBUJO puede ser grabado (SAVE) en cinta. Todas las instrucciones se trasladan al programa mientras se mantiene el DIBUJO en la pantalla.

La FIGURA FG1 muestra un mapa de la PENINSULA IBERICA tal y como se obtiene mediante este programa. Las coordenadas de la línea de la costa han sido tomadas del MAPA RESUMEN de la RED NACIONAL DE CARRETERAS (MOPU, ESPAÑA).

FIGURA FG1. **Un mapa realizado utilizando el programa "DIBUJO" (DIGITISER)**

COMO UTILIZAR EL PROGRAMA

Cuando se ejecuta (RUN), el programa pregunta por las coordenadas x e y que se van a asignar al punto inferior izquierdo, y por la coordenada x que se va a asignar al punto situado más a la derecha del diagrama.

A partir de estos datos, el programa calcula un factor de escala para el dibujo y la coordenada y máxima, preguntando al usuario si está conforme con ellas. Si esto es así, entonces el factor de escala obtenido se aplica a todos los valores de coordenadas a introducir a continuación, realizándose cualquier valor de x o de y que esté fuera de los límites definidos previamente (fuera de RANGO).

El programa reproduce a escala los puntos y los sitúa en el diagrama, cubriendo en la dirección x los pixels del 1 al 254 y del 9 al 174 en la dirección y. Para evitar los errores de redondeo en los cálculos de PLOT y DRAW y su repercusión en la pantalla, el programa avanza pixel a pixel hacia la izquierda, la derecha y parte superior del diagrama.

Además se deja una línea en la parte inferior de la pantalla para que pueda comunicarse el usuario con el programa.

El programa le indica al usuario que debe seleccionar un número tal que rebase los límites permitidos para x. Este número debe ser utilizado en adelante por el usuario como un "NUMERO DE RETORNO" que hace que el programa vuelva al MENU PRINCIPAL. El programa también sugiere al usuario que debe elegir los valores iniciales de los colores de BORDER (parte exterior de la pantalla), PAPER (parte interior de la pantalla) e INK (caracteres escritos sobre PAPER).

A diferencia de la mayoría de los programas de este libro, el presente programa no tiene salida por pantalla del MENU PRINCIPAL debido a que podría sobreimpresionarse sobre el contenido de la pantalla, que es el objetivo del programa. Toda comunicación entre el programa y el usuario se produce por medio de la línea de INPUT (la línea inferior de la pantalla). De esta manera, los mensajes del programa son algo ininteligibles.

La Tabla TG1 relaciona las INSTRUCCIONES que puede introducir el usuario (siempre mediante COMANDOS de una sola letra). Una vez pulsado un comando el programa salta a la subrutina precisa mediante la línea 1030.

INSTRUCCION	COMANDO	COMENTARIO
DIBUJO (DRAW)	D	Utilizado para introducir pares de coordenadas x e y para crear el DIBUJO de una línea.
COLOR	C	Coloración inicial para toda o parte del área de pantalla dentro (o fuera) de la imagen de línea.
INK Y PAPER	I	Nuevos valores de INK y PAPER para ser utilizados en adelante.
CARACTERIS-TICAS	A	Colores de INK y PAPER de las sentencias PRINT AT dadas previamente.
PALABRAS	W	Situar un mensaje en una posición dada.
RETOQUES	T	PLOT OVER 1 en una posición dada.
GRABAR (SAVE)	S	Grabar un cassette el contenido actual de la pantalla.
CARGAR (LOAD)	L	Cargar la "pantalla" deseada de cinta.
IMPRIMIT (PRINT)	P	Copiar el contenido de la pantalla en la impresora (PRINTER).
PARAR	H	Parar el programa.

TABLA TG1. **Las instrucciones que pueden ser introducidas por el usuario**

DIBUJO (DRAW) Introducir D

Esta es una subrutina con la cual se producen las imágenes de líneas que definen el diagrama. El programa comprueba que las parejas de coordenadas introducidas por el usuario tienen valores permitidos, calcula la escala adecuada y dibuja la línea correspondiente calculada en la pantalla. Los pares de coordenadas pueden también obtenerse mediante la impresora si está conectada. El usuario puede continuar introduciendo parejas de coordenadas hasta el momento en el que introduzca el "NUMERO DE RETORNO" para volver al MENU principal.

El comando DRAW OVER 1 sólo se utiliza si se comete algún error. Repitiendo la instrucción se borra la línea.

COLOR **Introducir C**

Esta opción se utiliza para colorear pixels (PLOT) con el valor actual del color de INK en toda o parte de la pantalla situada en la parte interior de la imagen de línea. El usuario especifica una "CAJA" que desea colorear y, después de comprobar que la "caja" no está situada fuera del área de pantalla el programa colorea (PLOT) dicha "caja" en la pantalla. El procedimiento de coloración comienza en la esquina inferior izquierda de la "caja" y continúa a lo largo de las sucesivas columnas de pixels.

El programa pregunta si debe conectarse o no este comando desde el principio y, cada vez, la posición actual de PLOT modifica la imagen preexistente.

la atención del usuario está polarizada por la posición actual de PLOT mediante la utilización del comando FLASH 1.

INK y PAPER **Introducir I**

Esta opción cambia los valores actuales de INK y PAPER.

CARACTERISTICAS, PALABRAS Y RETOQUES **Introducir A, W, T**

Estas "CARACTERISTICAS" específicas se refieren a los valores de INK y PAPER, y "PALABRAS" sitúa un mensaje (con valores INK y PAPER especificados previamente) en una posición PRINT AT dada.

"RETOQUES" dibuja (PLOT) un punto en un par dado de coordenadas.

En cada caso se puede utilizar el comando OVER 1, de tal forma que, si se repite la instrucción se borra el resultado. El "NUMERO DE RETORNO" se utiliza para volver al MENU principal.

PROGRAMA PG1. DIBUJO (DIGITISER)

```
 120 PRINT TAB 11;"Digitiser",,,
     "Introduce las coordenadas de la esquina inferior izquierda"
 130 PRINT ,,"Esquina inferior izquierda  = ";
 140 INPUT x1
 150 PRINT x1;",";
 160 INPUT y1
 170 PRINT y1
 180 PRINT ,,"Introduce la coordenada X de la esquina superior
     derecha"
 190 INPUT xu: IF xu<=x1 THEN  BEEP .2,24: GO TO 190
 200 LET yu=(xu-x1)*165/253-y1
 210 PRINT ,,"Esquina superior derecha = ";xu;",";yu
 220 PRINT ,,"Esta todo bien (S o N) ? "
 230 INPUT z$: IF z$="N" OR z$="n" THEN  RUN
 240 PRINT ,,"Elige un numero que no este entre  ";x1;" y ";xu;
     ".": PRINT "Este numero se usara para volveral MENU ."
 250 INPUT rx: IF rx>=x1 AND rx<=xu THEN  BEEP .2,24: GO TO 250
 310 PRINT "Border = ";
 320 INPUT b: IF b<0 OR b>7 THEN  BEEP .2,24: GO TO 320
 330 PRINT b
 340 PRINT "Paper  = ";
 350 INPUT p: IF p<0 OR p>7 THEN  BEEP .2,24: GO TO 350
 360 PRINT p;TAB 0;"Ink    = ";
 380 INPUT i: IF i<0 OR i>7 THEN  BEEP .2,24: GO TO 380
 390 PRINT i
 400 BORDER b: PAPER p: INK i
 990 CLS
1010 INPUT "MENU - elige opcion ";z$
1020 LET y$=CHR$ (CODE z$-32*(z$>"£"))
1030 GO TO 1100+4400*(y$="A")+4900*(y$="W")+3900*(y$="T")+3400*
     (y$="L")+400*(y$="C")+900*(y$="I")+1400*(y$="D")+1900*
     (y$="S")+2400*(y$="H")+2900*(y$="P")
1100 GO TO 1000
1500 INPUT "Color-esquina inferior izda ";x1b;",";y1b
1510 IF x1b<x1 OR x1b>xu OR y1b<y1 OR y1b>yu THEN  BEEP .2,24:
     GO TO 1500
1520 INPUT "Color-esquina superior dcha ";xub;",";yub
1530 IF xub<x1b+2 OR xub>xu OR yub<y1b+2 OR yub>yu THEN
     BEEP .2,24: GO TO 1520
1540 LET x1b=x1b*253/(xu-x1)+1
1550 LET xub=xub*253/(xu-x1)+1
1560 LET y1b=y1b*165/(yu-y1)+9
1570 LET yub=yub*165/(yu-y1)+9
1580 GO SUB 9000
1590 INPUT "Color del rectangulo correcto ? (S o N) ? ";z$
1600 IF z$<>"N" AND z$<>"n" THEN  GO TO 1630
1610 GO SUB 9000
1620 GO TO 1500
1650 INPUT "Color - si o no (s o n)?";z$
1660 IF z$<>"s" AND z$<>"S" AND z$<>"n" AND z$<>"N" THEN
     BEEP .2,24: GO TO 1650
1700 FOR j=x1b+1 TO xub-1
1710 LET y=(z$="s"): LET op=0
1720 FOR k=y1b+1 TO yub
1730 LET q=POINT (j,k)
1740 IF q=0 AND op=1 THEN  GO SUB 1800
1750 IF y=1 THEN  PLOT  INK i; PAPER p;j,k
1755 LET op=q:
1760 NEXT k
1770 NEXT j
1780 GO SUB 9000
```

```
1790 GO TO 1000
1805 PLOT  INK i; FLASH 1; PAPER p; OVER 1;j,k
1810 BEEP .2,24
1820 INPUT "Color - si o no (s o n)?";y$
1830 IF y$<>"s" AND y$<>"S" AND y$<>"n" AND y$<>"N" THEN
     BEEP .2,24: GO TO 1820
1840 LET y=(y$="s"): LET op=0
1850 PLOT  INK i; FLASH 0; PAPER p; OVER 1;j,k
1860 RETURN
2000 INPUT "Introduce el nuevo color del    fondo";p
2010 IF p<0 OR p>7 THEN  BEEP .2,24: GO TO 2000
2020 INPUT "Introduce el nuevo color de tin-ta";i
2030 IF i<0 OR i>7 THEN  BEEP .2,24: GO TO 2020
2040 GO TO 1000
2500 LPRINT : LPRINT "X  coordenadas  Y"
2510 INPUT "Draw -origen ?";ox;",";oy
2520 IF ox=rx THEN  GO TO 1100
2530 IF ox<xl OR ox>xu OR oy<yl OR oy>yu THEN
     BEEP .2,24: GO TO 2510
2535 LPRINT ox,oy
2540 LET ox=ox*253/(xu-xl)+1
2550 LET oy=oy*165/(yu-yl)+9
2560 PLOT  OVER 1; INK i;ox,oy
2600 INPUT "Draw - coordenadas ? ";x;",";y
2610 IF x=rx THEN  GO TO 1100
2620 IF x<xl OR x>xu OR y<yl OR y>yu THEN  BEEP .2,24: GO TO 2600
2625 LPRINT x,y
2630 LET x=x*253/(xu-xl)+1
2640 LET y=y*165/(yu-yl)+9
2650 LET px=x-ox
2660 LET py=y-oy
2670 DRAW  OVER 1; INK i;px,py
2680 LET ox=x
2690 LET oy=y
2700 GO TO 2600
3000 INPUT "Grabacion - nombre ?";z$
3010 IF LEN z$=0 OR LEN z$>10 THEN  BEEP .2,24: GO TO 3000
3020 SAVE z$SCREEN$
3030 GO TO 1000
3500 STOP
4000 COPY
4010 GO TO 1000
4500 INPUT "Carga - nombre ?";z$
4510 IF LEN z$>10 THEN  BEEP .2,24: GO TO 4500
4520 LOAD z$SCREEN$
4530 GO TO 1000
5000 INPUT "Retocado - coordenadas ";x;",";y
5010 IF x=rx THEN  GO TO 1000
5020 LET x=x*253/(xu-xl)+1
5030 LET y=y*165/(yu-yl)+9
5040 PLOT  OVER 1; INK i; PAPER p;x,y
5050 GO TO 5000
5500 INPUT "Attr - introduce coord. de AT ";r;",";c
5510 IF r=rx THEN  GO TO 1000
5520 IF r>21 OR r<0 OR c>31 OR c<0 THEN  BEEP .2,24: GO TO 5500
5530 PRINT  INK i; PAPER p; OVER 1;AT r,c;" ";
5540 GO TO 5500
6000 INPUT "Palabras - coordenadas de AT ";r;" ";c
6010 IF r=rx THEN  GO TO 1000
6020 IF r>21 OR r<0 OR c>31 OR c<0 THEN  BEEP .2,24: GO TO 6000
6030 INPUT "Palabras ?";z$
6040 PRINT  INK i; PAPER p; OVER 1;AT r,c;z$
6050 INPUT "Correcto ? (S or N) ? ";y$
```

```
6060 IF y$="N" OR y$="n" THEN  PRINT  INK i; PAPER p;
     OVER 1;AT r,c;z$
6070 GO TO 6000
9000 PLOT  INK i; PAPER p; OVER 1;xlb,ylb
9010 DRAW  INK i; PAPER p; OVER 1;xub-xlb,0
9020 DRAW  INK i; PAPER p; OVER 1;0,yub-ylb
9030 DRAW  INK i; PAPER p; OVER 1;xlb-xub,0
9040 DRAW  INK i; PAPER p; OVER 1;0,ylb-yub+1
9050 RETURN
```

20. CALEIDOSCOPIO

Completo mi colección de los 20 mejores programas para el ZX SPECTRUM presentando "CALEIDOSCOPIO". Es un programa simple, pero que garantiza la "hipnosis" del confiado usuario.

El programa trabaja cargando (POKE) una secuencia de números variable en el fichero de características y reflejando los valores obtenidos en 8 cuadrantes. La secuencia se genera por la ecuación de la línea 70. Podrían utilizarse otras ecuaciones, pero la que he utilizado proporciona una combinación satisfactoria de estabilidad y variación. Las líneas 80 a 150 cargan (POKE) el valor generado de K en los 8 cuadrantes, siendo cada uno de los cuadrantes una imagen aproximada en espejo de los cuadrantes vecinos.

PROGRAMA PQ1. CALEIDOSCOPIO

```
 10 LET d=0
 20 LET s=22928
 30 LET l=0
 40 FOR i=1 TO 12
 50 FOR j=0 TO 1
 60 LET d=d+.03
 70 LET k=INT (127*SIN d)
 80 POKE s+j,k
 90 POKE s-j-1,k
100 POKE s+32-31*i+32*j,k
110 POKE s-31*i-32*j,k
120 POKE s-33*i+31+32*j,k
130 POKE s-33*i-1-32*j,k
140 POKE s-64*i+31-j,k
150 POKE s-64*i+32+j,k
160 NEXT j
170 LET l=l+1
180 LET s=s+32
190 NEXT i
200 GO TO 20
```

FE DE ERRATAS

DIARIO

Página 24

```
410 PRINT ,,TAB 0; PAPER 6;"Introduce el dia, mes  y  ano  ";
     CHR$ 8;CHR$ 8;CHR$ 8;CHR$ 8; OVER 1;"_"

480 LET n$(i)=CHR$ (CODE n$(i)-32*(n$(i)>"£"))

550 PRINT ,,;"Ano  ";CHR$ 8;CHR$ 8;CHR$ 8;CHR$ 8; OVER 1;
    "_    1 ",
```

CALENDARIO

Páginas 28 y 29

```
30 INPUT z$: LET z$=CHR$ (CODE z$-32*(z$>"£")):
   IF z$<>"M" AND z$<>"A" THEN  BEEP 1,17: GO TO 30

50 INPUT c$: LET c$=CHR$ (CODE c$-32*(c$>"£")):
   IF c$<>"S" AND c$<>"N" THEN  BEEP 1,17: GO TO 50

540 LET n$(i)=CHR$ (CODE n$(i)-32*(n$(i)>"£"))
```

indescomp